```
                ****************************************
                *                                      *
                *       GMXBUG  SYSTEM  MONITOR        *
                *                                      *
                *       BY RICHARD C. ROSTROM          *
                *                                      *
                *       COPYRIGHT (C) 1979 BY          *
                *                                      *
                *       GIMIX, INC.                    *
                *       1337 W 37TH PLACE              *
                *       CHICAGO ILLINOIS 60609         *
                *                                      *
                ****************************************


                * SYSTEM SYMBOL DEFINITION
                *
                * SYMBOLIC EQUATIONS
                * ASCII CHARS
0004            EOT      EQU   $04
0007            BELL     EQU   $07
0008            BS       EQU   $08
000C            FF       EQU   $0C
000A            LF       EQU   $0A
000D            CR       EQU   $0D
001B            ESC      EQU   $1B
0020            SPACE    EQU   $20
002A            STAR     EQU   $2A        ASSEMBLER WILL NOT PROCESS
002F            SLASH    EQU   $2F        " #'* " OR " #'/ "
00FF            BLOCK    EQU   $FF


                * VIDEO SCREEN DEFINITION
F000            SCREEN   EQU   $F000
0050            LINE     EQU   80         80 CHARACTERS/LINE
0018            HEIGHT   EQU   24
                *
F780            ENDSCN   EQU   HEIGHT*LINE+SCREEN
F730            LOWLFT   EQU   ENDSCN-LINE
                *
                * VIDEO BOARD CONTROL REGISTERS
F900            CONTRL   EQU   $F900
F901            MODE     EQU   $F901
F901            SCROLL   EQU   $F901
F902            XCRSR    EQU   $F902
F903            YCRSR    EQU   $F903


                * I/O PORT ADDRESSES
8000            IOBLK    EQU   $8000
                * KEYBOARD PIA IS PORT #4
8010            KBDDAT   EQU   $8010
8011            KBDCTL   EQU   $8011
                * PARALLEL PRINTER PIA IS B SIDE PORT #4
8012            PPDATA   EQU   $8012
8013            PPCTL    EQU   $8013
                * CASSETTE ACIA IS PORT #0
```

```
8000          CASCTL  EQU    $8000
8001          CASDAT  EQU    $8001
              * SERIAL PRINTER ACIA IS PORT #3
800C          SPCTL   EQU    $800C
800D          SPDATA  EQU    $800D

              * ADDRESSES OF INTERRUPT PROCESSORS
E000          IRQENT  EQU    $E000
E005          NMIENT  EQU    $E005

              * SYSTEM CALL CODE ASSIGNMENTS
              * REGISTER AND ARITHMETIC FUNCTIONS
0000          XBA     EQU    0
0001          XBAX    EQU    1
0002          TXBA    EQU    2
0003          TBAX    EQU    3
0004          ADDAX   EQU    4
0005          ADDBAX  EQU    5
0006          SUBAX   EQU    6
0007          SUBBAX  EQU    7
0008          MULTAB  EQU    8
0009          PSHALL  EQU    9
000A          PULALL  EQU    10
000B          STAALL  EQU    11
              * MISCELLANEOUS
000C          HEXBIN  EQU    12
000D          BINHEX  EQU    13
000E          MOVER   EQU    14
000F          RESET   EQU    15
              * I/O FUNCTIONS
0010          INCHAR  EQU    16
0011          OUTCHR  EQU    17
0012          TAPEIN  EQU    18
0013          TAPOUT  EQU    19
0014          PSTRNG  EQU    20
0015          PSPACE  EQU    21
0016          PCRLF   EQU    22
0017          PFFEED  EQU    23
              * DEBUGGING FUNCTIONS
0018          BREAK0  EQU    24
0019          BREAK1  EQU    25
001A          BREAK2  EQU    26
001B          BREAK3  EQU    27
001C          PREGS   EQU    28
001D          PBYTE   EQU    29
001E          HEXIN   EQU    30
001F          DEBUGR  EQU    31
              * MISCELLANEOUS FUNCTIONS
0020          LINEIN  EQU    32
0021          MKWRIT  EQU    33
0022          MKREAD  EQU    34
0023          PCR     EQU    35
0024          PLF     EQU    36
0025          INECHO  EQU    37
0026          INKEY   EQU    38
0027          PRINT   EQU    39
0028          TBSRCH  EQU    40
```

```
0029      ESCSND  EQU    41
002A      INCAPS  EQU    42
          *
          * 43-45 TO BE FILLED IN LATER
          *

          * CODES FOR SPECIAL ESCAPE FUNCTIONS
0001      POSX    EQU    1
0002      POSY    EQU    2
0003      MEMOFF  EQU    3
0004      MEMON   EQU    4
0005      SLCVID  EQU    5
0006      SLCFNT  EQU    6
0007      SETM0   EQU    7
0008      SETM1   EQU    8
0009      VDBLNK  EQU    9
000A      VDNORM  EQU    10
000B      CBOFF   EQU    11
000C      CBON    EQU    12
000D      CBSTDY  EQU    13
000E      CBFLSH  EQU    14
000F      OUTSCR  EQU    15
0010      OUTPRT  EQU    16
0011      OUTBTH  EQU    17
0012      SLOT0   EQU    18
0013      SLOT1   EQU    19
0014      PRGMOD  EQU    20
0015      TRMODE  EQU    21

          * MONITOR MEMORY USAGE
          *
A000              ORG    $A000
          *
A000      SCRPAD  EQU    *
          *
          * INTERRUPT REQUEST VECTOR
A000      IRQVEC  RMB    2
          *
          * GENERAL PURPOSE
A002      BEGIN1  RMB    2
A004      END1    RMB    2
          *
          * NON-MASKABLE INTERRUPT VECTOR
A006      NMIVEC  RMB    2
          *
          * MIKBUG EQUIVALENCE
A008      SP      RMB    2
          *
A00A      SUBTOT  EQU    *
          *
A00A      CKSM    RMB    1
          *
A00B      BYTECT  RMB    1
A00C      XHI     RMB    1
A00D      XLO     RMB    1
A00E      MBTEMP  RMB    1
          *
```

```
A00F         TEMP    EQU    *
             *
A00F         TW      RMB    2
A011         MCONT   RMB    1
A012         XTEMP   RMB    2
             *
             * GENERAL PURPOSE
A014         BEGIN2  RMB    2
A016         END2    RMB    2


             * VIDEO DRIVER
A018         CURSOR  RMB    2
A01A         WRAP    RMB    1
A01B         CHRSLT  RMB    1
A01C         ESCSTR  RMB    1
             * LINE INPUT
A01D         ENDLIN  RMB    2
A01F         LINBUF  RMB    2
             * DEBUGGER STORAGE
A021         MEMPTR  RMB    2
A023         MMODE   RMB    1
             *
             * BREAKPOINT TABLE
A024         BRKTBL  RMB    16
             *
             * USER CODE VECTOR FOR SOFTWARE INTERRUPT
A034         USRVEC  RMB    2
             * OUTPUT DEVICE FLAG: $00=SCREEN ONLY, $01=BOTH,
             * $FF=PRINTER ONLY
A036         OUTPTR  RMB    1
             * # OF NULLS TO FOLLOW CR FOR PRINT DRIVER
A037         NULCNT  RMB    1
             * HALF/FULL DUPLEX FLAG  (1/0)
A038         DUPLEX  RMB    1
             * CTL-S SWITCH: IF 0, CTL-S IS DISABLED
A039         WAITON  RMB    1
             * FORCE UPPER CASE ALPHAS FLAG
A03A         UPCASE  RMB    1
             * VECTOR TO ACTIVE ESCAPE SEQUENCE ROUTINE
A03B         ESCVEC  RMB    2
             * SWI CONTROL: 0=GMXBUG TABLE  1=ADDRESS IN USRVEC
A03D         SWIFLG  RMB    1
             * PARALLEL/SERIAL PRINTER DRIVER (0/1)
A03E         PRFLAG  RMB    1
             * DATA FOR THE BELL ROUTINE
A03F         DURATN  RMB    2
A041         PERIOD  RMB    1


A048         PCLOCN  EQU    $A048


E191                 ORG    $E191


             * MONITOR CALL PROCESSOR
             * USES THE BYTE IMMEDIATELY FOLLOWING THE PC AS THE
             * CODE FOR ROUTINE BEING CALLED
```

```
E191 7D A0 3D  SWIENT  TST    SWIFLG     TEST FOR GMXBUG CONTROL OF SWI
E194 2A 05             BPL    MYCODE
E196 FE A0 34  USRCOD  LDX    USRVEC
E199 6E 00             JMP    X          JUMP TO USER-DEFINED ADDRESS
E19B 30        MYCODE  TSX
E19C A6 00             LDA A  X          GET STACKED CCR
E19E 06                TAP               RESTORE CCR
E19F EE 05             LDX    5,X        GET STACKED PC
E1A1 A6 00             LDA A  X          GET CODE BYTE
E1A3 81 2D             CMP A  #45        TEST FOR CODES OUT OF RANGE
E1A5 22 EF             BHI    USRCOD
E1A7 5F                CLR B
E1A8 48                ASL A             CALCULATE TABLE ENTRY ADDRESS
E1A9 7E E1 D8          JMP    SWIO       SKIP AROUND MIKBUG INEEE AND OUTEEE


E1D8                   ORG    $E1D8


E1D8 8B F7     SWIO    ADD A  #SWITBL
E1DA C9 E1             ADC B  #SWITBL/256
E1DC 36                PSH A             STORE ADDRESS ON STACK
E1DD 37                PSH B
E1DE 30                TSX
E1DF EE 00             LDX    X          INDEX GETS ADDRESS FORM STACK
E1E1 E6 00             LDA B  X          GET ROUTINE ADDRESS FROM TABLE
E1E3 A6 01             LDA A  1,X
E1E5 31                INS
E1E6 31                INS
E1E7 36                PSH A
E1E8 37                PSH B             STORE ROUTINE ADDRESS ON STACK
E1E9 30                TSX
E1EA A6 04             LDA A  4,X        RELOAD REGISTERS FROM STACK
E1EC E6 03             LDA B  3,X
E1EE 6C 08             INC    8,X        INCREMENT STACKED PC
E1F0 26 02             BNE    SWI1
E1F2 6C 07             INC    7,X
E1F4 EE 05     SWI1    LDX    5,X
E1F6 39                RTS               GO TO ADDRESS ON TOP OF STACK
               *
               * TABLE OF VECTORS
E1F7           SWITBL  EQU    *
E1F7 E2 60             FDB    XBAO
E1F9 E2 53             FDB    XBAXO
E1FB E2 5B             FDB    TXBAO
E1FD E2 6C             FDB    TBAXO
               *
E1FF E2 66             FDB    ADDAXO
E201 E2 67             FDB    ADDBAO
E203 E2 72             FDB    SUBAXO
E205 E2 73             FDB    SUBBAO
               *
E207 E2 80             FDB    MULTAO
E209 E2 95             FDB    PSHALO
E20B E2 B1             FDB    PULALO
E20D E2 CC             FDB    STAALO
               *
E20F E2 D8             FDB    HEXBIO
E211 E3 09             FDB    BINHEO
```

```
E213 E3 20                FDB    MOVERO
E215 E3 97                FDB    RESETO
                 *
E217 E1 29                FDB    $E129
E219 E9 85                FDB    OUTCHO
E21B E1 69                FDB    $E169
E21D E1 86                FDB    $E186
                 *
E21F EB F9                FDB    PSTRNO
E221 E9 7F                FDB    PSPACO
E223 E9 79                FDB    PCRLFO
E225 E9 83                FDB    PFFEEO
                 *
E227 E7 7C                FDB    BRKPRT     BREAKO
E229 E7 7C                FDB    BRKPRT     BREAK1
E22B E7 7C                FDB    BRKPRT     BREAK2
E22D E7 7C                FDB    BRKPRT     BREAK3
                 *
E22F E7 94                FDB    PREGSO
E231 E0 DB                FDB    $E0DB
E233 E0 E5                FDB    $E0E5
E235 E3 EE                FDB    DBGENT
                 *
E237 E0 14                FDB    $E014
E239 E8 61                FDB    MKWRIO
E23B E8 FE                FDB    MKREAO
E23D E9 75                FDB    PCRO
                 *
E23F E9 7B                FDB    PLFO
E241 E1 25                FDB    $E125
E243 E1 4C                FDB    $E14C
E245 E0 90                FDB    $E090
                 *
E247 E1 B5                FDB    $E1B5
E249 E1 43                FDB    $E143
E24B E1 7C                FDB    $E17C


                 * SPACE FOR 3 ADDITIONAL VECTORS
E24D E3 16                FDB    OPEN,OPEN,OPEN
                 *
                 * UTILITY SUBROUTINE PACKAGE
                 *
                 * REGISTER-TO-REGISTER OPERATIONS
                 *
                 * XBAX   #1
                 * EXCHANGE ACCUMULATORS AND INDEX
                 * IX <--> ACCB,ACCA
E253 3F          XBAXO    SWI
E254 02                   FCB    TXBA       MOVE IX TO ACCUMULATORS
E255 30                   TSX
E256 EE 01                LDX    1,X        LOAD IX WITH OLD ACCUMULATORS
E258 3F                   SWI
E259 0B                   FCB    STAALL     STORE IN STACK
E25A 3B                   RTI


                 * TXBA   #2
                 * MOVE INDEX TO ACCUMULATORS
```

```
                    * ACCB,ACCA <-- IX
E25B 30             TXBAO    TSX
E25C A6 03                   LDA A  3,X       LOAD ACCUMULATORS WITH INDEX
E25E E6 04                   LDA B  4,X
                    *
                    * XBA    #0
                    * EXCHANGE ACCUMULATORS
E260 30             XBAO     TSX              RELOAD IX FOR XBA ENTRY
E261 E7 02                   STA B  2,X
E263 A7 01                   STA A  1,X
E265 3B                      RTI


                    * ADDAX   #4
                    * ADD ACCA TO INDEX
                    * IX <-- IX + ACCA
E266 5F             ADDAXO   CLR B            ACCB <-- 0 THEN DO ADDITION
                    *
                    * ADDBAX  #5
                    * ADD ACCB,ACCA TO INDEX
                    * IX <-- IX + ACCB,ACCA
E267 30             ADDBAO   TSX
E268 AB 04                   ADD A  4,X
E26A E9 03                   ADC B  3,X
                    *
                    * TBAX    #3
                    * MOVE ACCB,ACCA TO INDEX
                    * IX <-- ACCB,ACCA
E26C 30             TBAXO    TSX              RELOAD IX FOR TBAX ENTRY
E26D E7 03          STXENT   STA B  3,X       STORE ACCUMULATORS IN STACKED IX
E26F A7 04                   STA A  4,X
E271 3B                      RTI


                    * SUBAX   #6
                    * SUBTRACT ACCA FROM INDEX
                    * IX <-- IX - ACCA
E272 5F             SUBAXO   CLR B            ACCB <-- THEN DO SUBTRACTION
                    *
                    * SUBBAX  #7
                    * SUBTRACT ACCB,ACCA FROM INDEX
                    * IX <-- IX - ACCB,ACCA
E273 36             SUBBAO   PSH A            STORE ACCUMULATORS ON TOP OF STACK
E274 37                      PSH B
E275 3F                      SWI
E276 02                      FCB    TXBA      MOVE INDEX VALUE TO ACCUMULATORS
E277 30                      TSX
E278 A0 01                   SUB A  1,X       SUBTRACT ACCUMULATOR VALUES
E27A E2 00                   SBC B  X
E27C 31                      INS              POP STACK FOR STORED ACCUMULATORS
E27D 31                      INS
E27E 20 EC                   BRA    TBAXO     GO STORE RESULT IN STACKED IX


                    * MULTAB  #8
                    * ACCB,ACCA <-- ACCA * ACCB
                    * UNSIGNED 8-BIT MULTIPLY ROUTINE
E280 86 08          MULTAO   LDA A  #8
E282 B7 A0 OF                STA A  TEMP      SET UP SHIFT COUNT
```

```
E285 30                    TSX
E286 4F                    CLR A
E287 56                    ROR B
E288 24 02       MULTA1    BCC    MULTA2    ADD ACCA FOR EACH SET BIT OF ACCB
E28A AB 02                 ADD A  2,X       TOTAL ACCUMULATES IN ACCA
E28C 46          MULTA2    ROR A            AND SHIFTS RIGHT INTO ACCB
E28D 56                    ROR B
E28E 7A A0 0F              DEC    TEMP      DECREMENT SHIFT COUNT
E291 26 F5                 BNE    MULTA1
E293 20 CB                 BRA    XBA0


                 * PSHALL  #9
                 * STORE CC, ACCB, ACCA, AND IX ON TOP OF STACK
E295 34          PSHAL0    DES              MAKE ROOM IN STACK FOR 5 BYTES
E296 34                    DES
E297 34                    DES
E298 34                    DES
E299 34                    DES
E29A 30                    TSX              COPY SP+5 THRU SP+11 TO
E29B C6 07                 LDA B  #7        SP THRU SP+6
E29D A6 05       PSHAL1    LDA A  5,X
E29F A7 00                 STA A  X
E2A1 08                    INX
E2A2 5A                    DEC B
E2A3 26 F8                 BNE    PSHAL1
E2A5 30                    TSX              COPY SP THRU SP+4 TO
E2A6 C6 05                 LDA B  #5        SP+7 THRU SP+11
E2A8 A6 00       PSHAL2    LDA A  X
E2AA A7 07                 STA A  7,X
E2AC 08                    INX
E2AD 5A                    DEC B
E2AE 26 F8                 BNE    PSHAL2
E2B0 3B                    RTI


                 * PULALL  #10
                 * LOAD CC, ACCB, ACCA, AND IX FROM STACK
E2B1 30          PULAL0    TSX              COPY SP+7 THRU SP+11 TO
E2B2 C6 05                 LDA B  #5        SP THRU SP+4
E2B4 A6 07       PULAL1    LDA A  7,X
E2B6 A7 00                 STA A  X
E2B8 08                    INX
E2B9 5A                    DEC B
E2BA 26 F8                 BNE    PULAL1
E2BC C6 07                 LDA B  #7        MOVE TOP 7 BYTES DOWN 5
E2BE A6 01       PULAL2    LDA A  1,X
E2C0 A7 06                 STA A  6,X
E2C2 09                    DEX
E2C3 5A                    DEC B
E2C4 26 F8                 BNE    PULAL2
E2C6 31                    INS              UPDATE STACK POINTER
E2C7 31                    INS
E2C8 31                    INS
E2C9 31                    INS
E2CA 31                    INS
E2CB 3B                    RTI
```

```
                    * STAALL  #11
                    * COPY ACCB, ACCA AND IX STORED BY CURRENT SWI TO
                    * THE REGISTER SAVE AREA OF THE PRECEDING SWI
E2CC 30             STAAL0  TSX
E2CD C6 04                  LDA B  #4          COPY 4 BYTES
E2CF A6 01          STAAL1  LDA A  1,X
E2D1 A7 08                  STA A  8,X
E2D3 08                     INX
E2D4 5A                     DEC B
E2D5 26 F8                  BNE    STAAL1
E2D7 3B                     RTI


                    * HEX-TO-BINARY AND BINARY-TO-HEX CONVERSION
                    *
                    * HEXBIN  #12
                    * CONVERTS A STRING POINTED TO BY IX TO 16-BIT BINARY
                    * CONVERSION TERMINATES ON 1ST NON-HEXADECIMAL CHAR
                    * RESULT RETURNED IN ACCB, ACCA
                    * IX POINTS TO TERMINATOR ON RETURN
E2D8 7F A0 0A       HEXBI0  CLR    SUBTOT      ZERO BINARY ACCUMULATOR
E2DB 4F                     CLR A
E2DC E6 00          HEXBI1  LDA B  X           GET A DIGIT
E2DE C0 30                  SUB B  #$30        CONVERT TO BINARY
E2E0 2D 22                  BLT    HEXBI4      TEST FOR <0
E2E2 C1 09                  CMP B  #9          TEST FOR >9
E2E4 2F 0A                  BLE    HEXBI2
E2E6 C1 11                  CMP B  #$11        TEST FOR >9, <A
E2E8 2D 1A                  BLT    HEXBI4
E2EA C1 16                  CMP B  #$16        TEST FOR >F
E2EC 2E 16                  BGT    HEXBI4
E2EE C0 07                  SUB B  #7          ADJUST FOR 9-A GAP
E2F0 48             HEXBI2  ASL A              SHIFT RESULT 4 BITS LEFT
E2F1 79 A0 0A               ROL    SUBTOT
E2F4 48                     ASL A
E2F5 79 A0 0A               ROL    SUBTOT
E2F8 48                     ASL A
E2F9 79 A0 0A               ROL    SUBTOT
E2FC 48                     ASL A
E2FD 79 A0 0A               ROL    SUBTOT
E300 1B                     ABA                INSERT 4 BIT DIGIT INTO RESULT
E301 08                     INX                MOVE STRING POINTER
E302 20 D8                  BRA    HEXBI1      NEXT DIGIT
E304 F6 A0 0A       HEXBI4  LDA B  SUBTOT      LOAD ACCB WITH MSB OF RESULT
E307 20 0B                  BRA    HEXBI5      GO STORE RESULT AND EXIT


                    * BINHEX  #13
                    * CONVERT BINARY ACCA TO 2 HEX ASCII DIGITS IN ACCB,ACCA
E309 36             BINHE0  PSH A              SAVE A COPY OF BINARY VALUE
E30A 8D 0B                  BSR    BCDHEX      CONVERT TO ASCII
E30C 16                     TAB                MOVE LS DIGIT TO ACCB
E30D 32                     PUL A              RECALL SAVED COPY
E30E 44                     LSR A              EXTRACT 4 MS BITS
E30F 44                     LSR A
E310 44                     LSR A
E311 44                     LSR A
E312 8D 03                  BSR    BCDHEX      CONVERT TO ASCII CHAR
                    *
```

```
E314 3F          HEXBI5  SWI               STORE RESULTS IN
E315 0B                  FCB     STAALL    STACKED REGISTERS
                 *
E316             ENDMOV  EQU     *
                 *
E316 3B          OPEN    RTI
                 *
E317 84 0F       BCDHEX  AND A   #%00001111 ELIMINATE 4 HIGH BITS
E319 8B 90               ADD A   #%10010000
E31B 19                  DAA
E31C 89 40               ADC A   #%01000000 CONVERT TO HEX
E31E 19                  DAA
E31F 39                  RTS


                 * MOVER    #14
                 * BEGIN1:  START ADDRESS FOR MOVE
                 * END1:    LAST BYTE TO MOVE
                 * BEGIN2:  START OF DESTINATION AREA
E320 FE A0 14    MOVER0  LDX     BEGIN2    LOAD ACCUMULATORS WITH DEST
E323 3F                  SWI
E324 02                  FCB     TXBA
E325 F1 A0 02            CMP B   BEGIN1    COMPARE DEST TO START
E328 26 05               BNE     MOVE1
E32A B1 A0 03            CMP A   BEGIN1+1  TEST FOR EQUALITY
E32D 27 E7               BEQ     ENDMOV    IF = NO MOVE NEEDED
E32F 25 0A       MOVE1   BCS     NMOVE     IF DEST<START MOVE NORMALLY
E331 F1 A0 04            CMP B   END1      COMPARE TO END
E334 26 03               BNE     MOVE2     IF DEST>END MOVE NORMALLY
E336 B1 A0 05            CMP A   END1+1
E339 23 1C       MOVE2   BLS     RMOVE     IF BEGIN1<=END1 REVERSE MOVE
E33B FE A0 02    NMOVE   LDX     BEGIN1    NORMAL MOVE (FRONT-TO-BACK)
E33E A6 00       NMOVE1  LDA A   X         LOAD BYTE
E340 FF A0 02            STX     BEGIN1    STORE SOURCE POINTER
E343 FE A0 14            LDX     BEGIN2    LOAD DEST POINTER
E346 A7 00               STA A   X         STORE BYTE
E348 08                  INX               BUMP POINTER
E349 FF A0 14            STX     BEGIN2    STORE DEST POINTER
E34C FE A0 02            LDX     BEGIN1    LOAD SOURCE POINTER
E34F BC A0 04            CPX     END1      TEST FOR END REACHED
E352 27 C2               BEQ     ENDMOV    QUIT ON END
E354 08                  INX               BUMP POINTER
E355 20 E7               BRA     NMOVE1    REPEAT WITH NEXT BYTE
E357 B0 A0 03    RMOVE   SUB A   BEGIN1+1  REVERSE MOVE (BACK-TO-FRONT)
E35A F2 A0 02            SBC B   BEGIN1    CALCULATE DEST+(END-START)
E35D BB A0 05            ADD A   END1+1
E360 F9 A0 04            ADC B   END1
E363 3F                  SWI
E364 03                  FCB     TBAX
E365 FF A0 14            STX     BEGIN2
E368 FE A0 04            LDX     END1      USE END POINTER FOR START
E36B A6 00       RMOVE1  LDA A   X
E36D FF A0 04            STX     END1
E370 FE A0 14            LDX     BEGIN2
E373 A7 00               STA A   X
E375 09                  DEX
E376 FF A0 14            STX     BEGIN2
E379 FE A0 04            LDX     END1
E37C BC A0 02            CPX     BEGIN1    EXIT WHEN SOURCE PTR=START
```

```
E37F 27 95            BEQ    ENDMOV
E381 09               DEX
E382 20 E7            BRA    RMOVE1

E384 1B      VDINIT   FCB    ESC,OUTSCR,ESC,PRGMOD,8
E389 00               FCB    $00,$10,$20,$30,$40,$50,$60,$70
E391 1B               FCB    ESC,SLOTO
E393 0C               FCB    FF,ESC,VDNORM,4

             * SYSTEM RESET ROUTINE
E397 CE 6F FF RESET0  LDX    #$6FFF     START STACK SEARCH AT 28K -1
E39A A6 00   RESET1   LDA A  X          FETCH THE BYTE
E39C 43               COM A
E39D 63 00            COM    X          COMPLEMENT MEMORY AND COPY
E39F A1 00            CMP A  X          STILL EQUAL?
E3A1 27 03            BEQ    RESET2     IF = RAM IS FOUND
E3A3 09               DEX
E3A4 20 F4            BRA    RESET1     GO TEST NEXT BYTE
E3A6 FF A0 1F RESET2  STX    LINBUF     SET LINBUF TO END OF MEMORY
E3A9 35               TXS               SET STACK THERE
E3AA 31               INS               FIX FOR -1 IN TXS
E3AB CE A0 00         LDX    #SCRPAD    SET UP PTR TO SCRATCH PAD RAM
E3AE 6F 3D            CLR    $3D,X      RESET SWI CONTROL FLAG
E3B0 6F 20            CLR    $20,X      SET LINBUF TO START OF LAST PAGE
E3B2 6F 1C            CLR    $1C,X      CLEAR ESCAPE SEQUENCE FLAG
E3B4 6F 3E            CLR    $3E,X      INITIALIZE FOR PARALLEL PRINTER
E3B6 6F 39            CLR    $39,X      TURN OFF CTL-S
E3B8 86 3C            LDA A  #60
E3BA A7 41            STA A  $41,X      INITIALIZE BELL ROUTINE PARAMS
E3BC A7 3F            STA A  $3F,X
E3BE 6F 40            CLR    $40,X
E3C0 86 06            LDA A  #6
E3C2 A7 37            STA A  $37,X      INITIALIZE NULCNT TO 6
E3C4 CE 80 00         LDX    #IOBLK     SET UP POINTER TO I/O AREA
E3C7 44               LSR A             RESET CASSETTE INTERFACE &
E3C8 A7 00            STA A  X          SERIAL PRINTER INTERFACE
E3CA A7 0C            STA A  $0C,X
E3CC 86 11            LDA A  #$11       SET TO 8-2-NONE
E3CE A7 00            STA A  X
E3D0 A7 0C            STA A  $0C,X
E3D2 86 2F            LDA A  #$2F
E3D4 A7 11            STA A  $11,X      INITIALIZE KEYBOARD INTERFACE
E3D6 A6 10            LDA A  $10,X
E3D8 6F 13            CLR    $13,X      INIT PARALLEL PRINTER INTERFACE
E3DA 86 FF            LDA A  #$FF
E3DC A7 12            STA A  $12,X
E3DE 86 3E            LDA A  #$3E
E3E0 A7 13            STA A  $13,X
E3E2 A6 12            LDA A  $12,X
E3E4 86 41            LDA A  #%01000001
E3E6 B7 F9 00         STA A  CONTRL
E3E9 CE E3 84         LDX    #VDINIT    INITIALIZE VIDEO DRIVER
E3EC 3F               SWI
E3ED 14               FCB    PSTRNG

             * DEBUGGER
```

```
E3EE 7F A0 23  DBGENT  CLR    MMODE      RETURN TO STD MODE
E3F1 7F A0 38          CLR    DUPLEX
E3F4 86 FF             LDA A  #$FF
E3F6 20 08             BRA    DEBO

               * INTERRUPT VECTORS
               *
E3F8 E0 00             FDB    IRQENT
E3FA E1 91             FDB    SWIENT
E3FC E0 05             FDB    NMIENT
E3FE E3 97             FDB    RESETO

E400 B7 A0 3A  DEBO    STA A  UPCASE
E403 3F        DEBUG   SWI
E404 16                FCB    PCRLF
E405 86 3E             LDA A  #'>        PRINT A ">"
E407 3F                SWI
E408 11                FCB    OUTCHR
E409 CE E4 45          LDX    #JTABLE
E40C 7D A0 23          TST    MMODE
E40F 27 0F             BEQ    DEB1
E411 FE A0 21          LDX    MEMPTR     IN MEMORY MODE, PRINT MEMPTR
E414 BD E5 C2          JSR    PRX
E417 A6 00             LDA A  X
E419 3F                SWI
E41A 1D                FCB    PBYTE
E41B 3F                SWI
E41C 15                FCB    PSPACE
E41D CE E4 30          LDX    #JTABL1
E420 3F        DEB1    SWI
E421 2A                FCB    INCAPS     INPUT A COMMAND CHARACTER
E422 85 E0             BIT A  #%11100000 TEST FOR CONTROL CHARS
E424 27 02             BEQ    NOECHO     IF CHAR IS CTL, DON'T ECHO
E426 3F                SWI
E427 11                FCB    OUTCHR     ECHO ALL PRINTING CHARS

E428 3F        NOECHO  SWI
E429 28                FCB    TBSRCH     GET ADDRESS FROM TABLE
E42A 26 D7             BNE    DEBUG
E42C AD 00             JSR    X          JSR TO ROUTINE
E42E 20 D3             BRA    DEBUG      GO TO GET NEXT COMMAND

E430           JTABL1  EQU    *
E430 2B                FCC    /+/
E431 E4 86             FDB    DOPLUS
E433 2D                FCC    /-/
E434 E4 8E             FDB    DOMINS
E436 3D                FCC    /=/
E437 E4 98             FDB    ASSIGN
E439 0D                FCB    CR
E43A E4 94             FDB    MMOFF
E43C 20                FCB    SPACE
E43D E4 82             FDB    ENTER
E43F 22                FCC    /"/
E440 E4 A1             FDB    ASCENT
```

```
E442 32                   FCC     /2/
E443 E4 BD                FDB     DO2
                *
E445            JTABLE    EQU     *
E445 41                   FCC     /A/
E446 E5 AE                FDB     ARITHM
E448 42                   FCC     /B/
E449 E7 C4                FDB     BREAK
E44B 43                   FCC     /C/
E44C E5 CE                FDB     CHKSUM
E44E 44                   FCC     /D/
E44F E5 F1                FDB     DUMP
E451 45                   FCC     /E/
E452 E0 52                FDB     $E052     ERASE SCREEN CMD IN MIKEXT AREA
E454 46                   FCC     /F/
E455 E6 2E                FDB     FMTDMP
E457 47                   FCC     /G/
E458 E6 75                FDB     GOTO
E45A 48                   FCC     /H/
E45B E7 35                FDB     HXLOCT
E45D 49                   FCC     /I/
E45E E4 D3                FDB     INITLZ
E460 4A                   FCC     /J/
E461 E6 93                FDB     JSRTO
E463 4B                   FCC     /K/
E464 E8 0B                FDB     KILBRK
E466 4C                   FCC     /L/
E467 E6 97                FDB     LOAD
E469 4D                   FCC     /M/
E46A E6 A6                FDB     MEMORY
E46C 50                   FCC     /P/
E46D E8 4B                FDB     PRTBRK
E46F 52                   FCC     /R/
E470 E7 98                FDB     RDUMP
E472 53                   FCC     /S/
E473 E6 AF                FDB     SAVE
E475 54                   FCC     /T/
E476 E6 E5                FDB     TSTMEM
E478 55                   FCC     /U/
E479 E0 61                FDB     $E061     USER PROM CMD IN MIKEXT AREA
E47B 58                   FCC     /X/
E47C E6 B4                FDB     XFER
E47E 5A                   FCC     /Z/
E47F E6 BE                FDB     ZAPMEM
E481 00                   FCB     0
E482            ENDTBL    EQU     *


E482 8D 14      ENTER     BSR     ASSIGN
E484 29 07                BVS     DOP2
                *
E486 FE A0 21   DOPLUS    LDX     MEMPTR
E489 08                   INX
E48A FF A0 21   DOP1      STX     MEMPTR
E48D 39         DOP2      RTS
                *
E48E FE A0 21   DOMINS    LDX     MEMPTR
E491 09                   DEX
E492 20 F6                BRA     DOP1
```

```
                    *
E494 7F A0 23       MMOFF   CLR     MMODE
E497 39                     RTS
                    *
E498 BD E6 DE       ASSIGN  JSR     GETNUM
E49B FE A0 21               LDX     MEMPTR
E49E A7 00                  STA A   X
E4A0 39                     RTS
                    *
E4A1 3F             ASCENT  SWI
E4A2 15                     FCB     PSPACE
E4A3 FE A0 21               LDX     MEMPTR
E4A6 3F             ASC1    SWI
E4A7 10                     FCB     INCHAR
E4A8 85 E0                  BIT A   #$E0        MASK OUT CTL CHARS
E4AA 27 07                  BEQ     ASC2
E4AC 3F                     SWI
E4AD 11                     FCB     OUTCHR
E4AE A7 00                  STA A   X
E4B0 08                     INX
E4B1 20 F3                  BRA     ASC1
E4B3 81 1B          ASC2    CMP A   #ESC
E4B5 27 04                  BEQ     ASC3
E4B7 81 0D                  CMP A   #CR
E4B9 26 EB                  BNE     ASC1
E4BB 20 CD          ASC3    BRA     DOP1
                    *
                    * DISPLAY CURRENT BYTE IN BASE 2
E4BD FE A0 21       DO2     LDX     MEMPTR
E4C0 E6 00                  LDA B   X
E4C2 8D 00                  BSR     DO20
E4C4 CE 00 04       DO20    LDX     #4
E4C7 3F                     SWI
E4C8 15                     FCB     PSPACE
E4C9 86 18          DO21    LDA A   #$18
E4CB 58                     ASL B
E4CC 49                     ROL A
E4CD 3F                     SWI
E4CE 11                     FCB     OUTCHR
E4CF 09                     DEX
E4D0 26 F7                  BNE     DO21
E4D2 39             INITL9  RTS


                    * GENERAL INITIALIZER
E4D3 8D 7D          INITLZ  BSR     CHARB       GET ID OF THING TO INIT
E4D5 CE E4 E0               LDX     #INTTBL     GET TABLE OF INIT THINGS
E4D8 3F                     SWI
E4D9 28                     FCB     TBSRCH
E4DA 26 F6                  BNE     INITL9
E4DC 3F             INITL2  SWI
E4DD 11                     FCB     OUTCHR      CHARACTER IS VALID; ECHO
E4DE 6E 00                  JMP     X
                    * TABLE OF INIT ROUTINES
E4E0 43             INTTBL  FCC     /C/
E4E1 E5 42                  FDB     INITCS
E4E3 44                     FCC     /D/
E4E4 E4 F9                  FDB     INITDX
E4E6 4E                     FCC     /N/
```

```
E4E7 E5 09            FDB    INITNL
E4E9 4F               FCC    /O/
E4EA E5 10            FDB    INITOP
E4EC 50               FCC    /P/
E4ED E5 25            FDB    INITPF
E4EF 52               FCC    /R/
E4F0 E5 58            FDB    INITRP
E4F2 53               FCC    /S/
E4F3 E5 36            FDB    INITSP
E4F5 55               FCC    /U/
E4F6 E3 97            FDB    INITUN
E4F8 00               FCB    0
                *
E397            INITUN  EQU    RESETO
                *
E4F9 8D 57      INITDX  BSR    CHARB
E4FB 81 46              CMP A  #'F
E4FD 27 05              BEQ    INITD1
E4FF 81 48              CMP A  #'H
E501 26 32              BNE    INITP2
E503 53                 COM B
E504 F7 A0 38   INITD1  STA B  DUPLEX
E507 20 2A              BRA    INITPX
                *
E509 BD E6 DE   INITNL  JSR    GETNUM
E50C B7 A0 37           STA A  NULCNT
E50F 39                 RTS
                *
E510 8D 40      INITOP  BSR    CHARB
E512 81 53              CMP A  #'S
E514 27 0A              BEQ    INITO1
E516 5A                 DEC B
E517 81 50              CMP A  #'P
E519 27 05              BEQ    INITO1
E51B 50                 NEG B
E51C 81 42              CMP A  #'B
E51E 26 15              BNE    INITP2
E520 F7 A0 36   INITO1  STA B  OUTPTR
E523 20 0E              BRA    INITPX
                *
E525 8D 2B      INITPF  BSR    CHARB
E527 81 50              CMP A  #'P
E529 27 05              BEQ    INITP1
E52B 53                 COM B
E52C 81 53              CMP A  #'S
E52E 26 05              BNE    INITP2
E530 F7 A0 3E   INITP1  STA B  PRFLAG
E533 3F         INITPX  SWI
E534 11                 FCB    OUTCHR
E535 39         INITP2  RTS
                *
E536 BD E6 DE   INITSP  JSR    GETNUM
E539 08                 INX
E53A 09                 DEX
E53B 27 F8              BEQ    INITP2
E53D 35                 TXS
E53E 31                 INS
E53F 7E E4 03           JMP    DEBUG
                *
```

```
E542 8D 0E      INITCS  BSR    CHARB
E544 81 4E              CMP A  #'N
E546 27 05              BEQ    INITC1
E548 81 59              CMP A  #'Y
E54A 26 E9              BNE    INITP2
E54C 53                 COM B
E54D F7 A0 39   INITC1  STA B  WAITON
E550 20 E1              BRA    INITPX
                *
E552 3F         CHARB   SWI
E553 15                 FCB    PSPACE
E554 3F                 SWI
E555 2A                 FCB    INCAPS
E556 5F                 CLR B
E557 39                 RTS
                *
E558 3F         INITRP  SWI
E559 15                 FCB    PSPACE
E55A C6 48              LDA B  #'H
E55C 7D A0 38           TST    DUPLEX
E55F 26 02              BNE    INITR1
E561 C6 46              LDA B  #'F
E563 86 44      INITR1  LDA A  #'D
E565 8D 3C              BSR    RPRT
E567 C6 53              LDA B  #'S
E569 7D A0 36           TST    OUTPTR
E56C 27 08              BEQ    INITR3
E56E 2A 04              BPL    INITR2
E570 C6 50              LDA B  #'P
E572 20 02              BRA    INITR3
E574 C6 42      INITR2  LDA B  #'B
E576 86 4F      INITR3  LDA A  #'O
E578 8D 29              BSR    RPRT
E57A C6 59              LDA B  #'Y
E57C 7D A0 39           TST    WAITON
E57F 26 02              BNE    INITR4
E581 C6 4E              LDA B  #'N
E583 86 53      INITR4  LDA A  #'S
E585 8D 1C              BSR    RPRT
E587 86 4E              LDA A  #'N
E589 3F                 SWI
E58A 11                 FCB    OUTCHR
E58B 86 3D              LDA A  #'=
E58D 3F                 SWI
E58E 11                 FCB    OUTCHR
E58F B6 A0 37           LDA A  NULCNT
E592 8D 35              BSR    PBYTSP
E594 C6 50              LDA B  #'P
E596 17                 TBA
E597 7D A0 3E           TST    PRFLAG
E59A 27 02              BEQ    INITR5
E59C C6 53              LDA B  #'S
E59E 8D 03      INITR5  BSR    RPRT
E5A0 3F                 SWI
E5A1 1C                 FCB    PREGS
E5A2 39                 RTS
E5A3 3F         RPRT    SWI
E5A4 11                 FCB    OUTCHR
E5A5 86 3D              LDA A  #'=
```

```
E5A7 3F                  SWI
E5A8 11                  FCB    OUTCHR
E5A9 17                  TBA
E5AA 3F                  SWI
E5AB 11                  FCB    OUTCHR
E5AC 20 1D               BRA    PSP

E5AE BD E6 CF  ARITHM    JSR    GETADS
E5B1 BB A0 03            ADD A  BEGIN1+1
E5B4 F9 A0 02            ADC B  BEGIN1
E5B7 3F                  SWI
E5B8 15                  FCB    PSPACE
E5B9 8D 09               BSR    PRBA
E5BB 3F                  SWI
E5BC 02                  FCB    TXBA
E5BD FE A0 02            LDX    BEGIN1
E5C0 3F                  SWI
E5C1 07                  FCB    SUBBAX
E5C2 3F        PRX       SWI
E5C3 02                  FCB    TXBA
E5C4 36        PRBA      PSH A
E5C5 17                  TBA
E5C6 3F                  SWI
E5C7 1D                  FCB    PBYTE
E5C8 32                  PUL A
E5C9 3F        PBYTSP    SWI
E5CA 1D                  FCB    PBYTE
E5CB 3F        PSP       SWI
E5CC 15                  FCB    PSPACE
E5CD 39                  RTS

E5CE BD E6 CF  CHKSUM    JSR    GETADS    GET 2 ADDRESSES
E5D1 4F                  CLR A            INIT CHECKSUM ACCUMULATOR
E5D2 5F                  CLR B
E5D3 7F A0 0A            CLR    SUBTOT
E5D6 08                  INX
E5D7 09        CHECK1    DEX
E5D8 AB 00               ADD A  X         ADD A BYTE TO THE CHECKSUM
E5DA C9 00               ADC B  #0        ADD THE CARRY THE MIDDLE BYTE
E5DC 24 03               BCC    CHECK2
E5DE 7C A0 0A            INC    SUBTOT    ADD CARRY IF ANY TO 3RD BYTE
E5E1 BC A0 02  CHECK2    CPX    BEGIN1    TEST FOR END OF SUM AREA
E5E4 26 F1               BNE    CHECK1
E5E6 3F        CHECK3    SWI
E5E7 15                  FCB    PSPACE
E5E8 36                  PSH A  SAVE      LSB OF SUM
E5E9 B6 A0 0A            LDA A  SUBTOT    PRINT MSB
E5EC 3F                  SWI
E5ED 1D                  FCB    PBYTE
E5EE 32                  PUL A
E5EF 20 D3               BRA    PRBA

E5F1 BD E6 DE  DUMP      JSR    GETNUM
E5F4 FF A0 02  DUMP1     STX    BEGIN1
E5F7 3F                  SWI
E5F8 16                  FCB    PCRLF
```

```
E5F9 8D C7                BSR    PRX
E5FB C6 10                LDA B  #16
E5FD A6 00       DUMP2    LDA A  X
E5FF 3F                   SWI
E600 15                   FCB    PSPACE
E601 3F                   SWI
E602 1D                   FCB    PBYTE
E603 08                   INX
E604 5A                   DEC B
E605 26 F6                BNE    DUMP2
E607 3F                   SWI
E608 15                   FCB    PSPACE
E609 3F                   SWI
E60A 15                   FCB    PSPACE
E60B 86 2F                LDA A  #SLASH
E60D 3F                   SWI
E60E 11                   FCB    OUTCHR
E60F 86 15                LDA A  #TRMODE
E611 3F                   SWI
E612 29                   FCB    ESCSND
E613 86 10                LDA A  #16
E615 3F                   SWI
E616 11                   FCB    OUTCHR
E617 16                   TAB
E618 FE A0 02             LDX    BEGIN1
E61B A6 00       DUMP3    LDA A  X
E61D 3F                   SWI
E61E 11                   FCB    OUTCHR
E61F 08                   INX
E620 5A                   DEC B
E621 26 F8                BNE    DUMP3
E623 86 2F                LDA A  #SLASH
E625 3F                   SWI
E626 11                   FCB    OUTCHR
E627 3F                   SWI
E628 10                   FCB    INCHAR
E629 81 1B                CMP A  #ESC
E62B 26 C7                BNE    DUMP1
E62D 39                   RTS


E62E BD E6 DE    FMTDMP   JSR    GETNUM
E631 3F          FORM1    SWI
E632 16                   FCB    PCRLF
E633 8D 8D                BSR    PRX
E635 3F                   SWI
E636 15                   FCB    PSPACE
E637 A6 00                LDA A  X         GET NEXT OP CODE BYTE
E639 C6 03                LDA B  #3        ACCB WILL BE COUNTER
E63B 81 8C                CMP A  #$8C      CHECK CPX IMM
E63D 27 26                BEQ    FORM3
E63F 81 8E                CMP A  #$8E      CHECK LDS IMM
E641 27 22                BEQ    FORM3
E643 81 CE                CMP A  #$CE      CHECK LDX IMM
E645 27 1E                BEQ    FORM3
E647 5A                   DEC B
E648 81 3F                CMP A  #$3F      CHECK SWI
E64A 27 19                BEQ    FORM3
E64C 84 F0                AND A  #$F0      DECODE BY 16S
```

```
E64E 5A                      DEC B
E64F 81 30                   CMP A    #$30
E651 27 12                   BEQ      FORM3
E653 C6 03                   LDA B    #3
E655 43                      COM A
E656 85 30                   BIT A    #%00110000
E658 27 0B                   BEQ      FORM3
E65A 43                      COM A
E65B 5A                      DEC B
E65C 81 20                   CMP A    #$20
E65E 27 05                   BEQ      FORM3
E660 81 50                   CMP A    #$50
E662 22 01                   BHI      FORM3
E664 5A                      DEC B
E665 A6 00          FORM3    LDA A    X           DISPLAY 1, 2 OR 3 BYTES ACROSS
E667 BD E5 C9                JSR      PBYTSP
E66A 08                      INX
E66B 5A                      DEC B
E66C 26 F7                   BNE      FORM3
E66E 3F                      SWI
E66F 10                      FCB      INCHAR
E670 81 1B                   CMP A    #ESC
E672 26 BD                   BNE      FORM1
E674 39                      RTS


E675 8D 67          GOTO     BSR      GETNUM      GET AN ADDRESS
E677 31                      INS
E678 31                      INS                  POP RETURN ADDRESS OFF STACK
E679 08                      INX
E67A 09                      DEX                  IF ADDR=0 MAY BE EXIT FROM BRKPT
E67B 26 14                   BNE      GOTO9
E67D FE A0 1F                LDX      LINBUF      GET 1ST BYTE OF LINE BUFFER
E680 A6 00                   LDA A    X
E682 81 58                   CMP A    #'X         TEST FOR "X"
E684 26 01                   BNE      GOTO3       IF NOT X CHECK FOR CR
E686 3B                      RTI                  EXIT FROM BREAKPOINT
E687 81 0D          GOTO3    CMP A    #CR         TEST FOR CR
E689 27 03                   BEQ      GOTO4       IF NOT CR GO TO 0000
E68B 7E 00 00                JMP      0           0 WAS ENTERED; JUMP TO 0
E68E FE A0 48       GOTO4    LDX      PCLOCN      GET ADDRESS IN PC OF OLD STACK
E691 6E 00          GOTO9    JMP      X           GO TO THAT ADDRESS


E693 8D 49          JSRTO    BSR      GETNUM      GET AN ADDRESS
E695 6E 00                   JMP      X           JUMP TO ADDRESS


E697 8D 45          LOAD     BSR      GETNUM
E699 4F                      CLR A                USE FLAG IS EITHER 00 OR FF
E69A FF A0 14                STX      BEGIN2      SET UP STARTING STORAGE ADDRESS
E69D 27 01                   BEQ      LOAD1
E69F 43                      COM A                SET USE FLAG TO "IGNORE"
E6A0 B7 A0 10       LOAD1    STA A    TEMP+1
E6A3 3F                      SWI
E6A4 22                      FCB      MKREAD
E6A5 39                      RTS
```

```
E6A6 8D 36       MEMORY  BSR    GETNUM
E6A8 FF A0 21            STX    MEMPTR
E6AB 7C A0 23            INC    MMODE
E6AE 39                  RTS


E6AF 8D 1E       SAVE    BSR    GETADS    GET BOUNDS OF MEMORY TO SAVE
E6B1 3F                  SWI
E6B2 21                  FCB    MKWRIT    GO WRITE DATA TO TAPE
E6B3 39                  RTS


                 * MOVE A BLOCK OF MEMORY
E6B4 8D 19       XFER    BSR    GETADS    GET START AND END ADDRESSES
E6B6 8D 26               BSR    GETNUM    GET DESTINATION ADDRESS
E6B8 FF A0 14            STX    BEGIN2
E6BB 3F                  SWI
E6BC 0E                  FCB    MOVER     DO MOVE
E6BD 39                  RTS


E6BE 8D 0F       ZAPMEM  BSR    GETADS    GET BOUNDS OF BLOCK TO FILL
E6C0 8D 1C               BSR    GETNUM    GET HEX VALUE TO STORE
E6C2 FE A0 02            LDX    BEGIN1    GET START OF MEMORY TO FILL
E6C5 09                  DEX
E6C6 08          ZAP1    INX
E6C7 A7 00               STA A  X
E6C9 BC A0 04            CPX    END1
E6CC 26 F8               BNE    ZAP1
E6CE 39                  RTS


E6CF 3F          GETADS  SWI
E6D0 1E                  FCB    HEXIN     GET THE FIRST ADDRESS
E6D1 29 0F               BVS    NOADS
E6D3 FF A0 02            STX    BEGIN1
E6D6 3F                  SWI
E6D7 1E                  FCB    HEXIN     GET THE SECOND ADDRESS
E6D8 29 08               BVS    NOADS
E6DA FF A0 04            STX    END1
E6DD 39                  RTS
                 *
E6DE 3F          GETNUM  SWI
E6DF 1E                  FCB    HEXIN
E6E0 28 02               BVC    GETN1
                 *
E6E2 31          NOADS   INS
E6E3 31                  INS
E6E4 39          GETN1   RTS


E6E5 8D E8       TSTMEM  BSR    GETADS    GET ADDRESSES
E6E7 3F                  SWI
E6E8 16                  FCB    PCRLF
E6E9 5F                  CLR B  ACCB      WILL COUNT 256 PASSES
E6EA FE A0 02    TEST1   LDX    BEGIN1    GET START OF TEST AREA
E6ED 09                  DEX
E6EE 08          TEST2   INX
E6EF 8D 28               BSR    MAKBYT    MAKE NEXT WORD IN SEQUENCE
```

```
E6F1 A7 00              STA A   X           WRITE INTO MEMORY
E6F3 BC A0 04           CPX     END1        TEST FOR END OF BLOCK
E6F6 26 F6              BNE     TEST2
E6F8 FE A0 02           LDX     BEGIN1
E6FB 09                 DEX
E6FC 08         TEST3   INX
E6FD 8D 1A              BSR     MAKBYT      MAKE BYTE FOR TEST
E6FF A8 00              EOR A   X           CHECK AGAINST MEMORY
E701 26 13              BNE     TEST6       NON-0 RESULT INDICATES FAILURE
E703 BC A0 04           CPX     END1
E706 26 F4              BNE     TEST3
                * THE END OF LINE TEST IS NEEDED BECAUSE OUTPUT
                * MAY GO TO A HARD COPY DEVICE
E708 C5 3F              BIT B   #%00111111 TEST FOR END OF LINE
E70A 26 02              BNE     TEST5
E70C 3F                 SWI
E70D 16                 FCB     PCRLF
E70E 86 23      TEST5   LDA A   #'#         PRINT A # FOR EACH PASS
E710 3F                 SWI
E711 11                 FCB     OUTCHR
E712 5C                 INC B   BUMP        AND TEST PASS COUNTER
E713 26 D5              BNE     TEST1
E715 39                 RTS
E716 3F         TEST6   SWI
E717 1C                 FCB     PREGS       IF ERROR FOUND DUMP REGS. IX
E718 39                 RTS                 HAS BAD LOCN, ACCA HAS BAD BITS
E719 FF A0 0A   MAKBYT  STX     SUBTOT      STORE IX IN A WORK LOCATION
E71C 17                 TBA                 RESULT IS PASS# + IXL + IXH
E71D BB A0 0A           ADD A   SUBTOT
E720 BB A0 0B           ADD A   SUBTOT+1
E723 39                 RTS


E724 3F         GETBYT  SWI
E725 1E                 FCB     HEXIN
E726 29 BA              BVS     NOADS
E728 7A A0 1E           DEC     ENDLIN+1
E72B 27 04              BEQ     GETB1
E72D 7C A0 12           INC     XTEMP
E730 39         FINDXT  RTS
E731 31         GETB1   INS
E732 31                 INS
E733 20 14              BRA     FIND05
                *
E735 8D 98      HXLOCT  BSR     GETADS
E737 7F A0 12           CLR     XTEMP
E73A 8D E8              BSR     GETBYT
E73C B7 A0 0F           STA A   TEMP
E73F 8D E3              BSR     GETBYT
E741 B7 A0 10           STA A   TEMP+1
E744 8D DE              BSR     GETBYT
E746 B7 A0 0E           STA A   MBTEMP
E749 FE A0 02   FIND05  LDX     BEGIN1
E74C F6 A0 12   FIND10  LDA B   XTEMP
E74F 27 DF              BEQ     FINDXT
E751 B6 A0 0F           LDA A   TEMP
E754 A1 00              CMP A   X
E756 26 14              BNE     FIND20
E758 5A                 DEC B
```

```
E759 27 19                  BEQ     FIND30
E75B B6 A0 10               LDA A   TEMP+1
E75E A1 01                  CMP A   1,X
E760 26 0A                  BNE     FIND20
E762 5A                     DEC B
E763 27 0F                  BEQ     FIND30
E765 B6 A0 0E               LDA A   MBTEMP
E768 A1 02                  CMP A   2,X
E76A 27 08                  BEQ     FIND30
E76C BC A0 04     FIND20    CPX     END1
E76F 27 BF                  BEQ     FINDXT
E771 08                     INX
E772 20 D8                  BRA     FIND10
E774 FF A0 21     FIND30    STX     MEMPTR
E777 3F                     SWI
E778 15                     FCB     PSPACE
E779 7E E5 C2               JMP     PRX


                  * ENTRANCE FROM BREAKPOINT
E77C 3F           BRKPRT    SWI
E77D 16                     FCB     PCRLF       START NEW LINE
E77E 86 24                  LDA A   #'$         PRINT "$$"
E780 3F                     SWI
E781 11                     FCB     OUTCHR
E782 3F                     SWI
E783 11                     FCB     OUTCHR
E784 30                     TSX     GET         PC FROM STACK
E785 EE 05                  LDX     5,X
E787 09                     DEX                 PREVIOUS BYTE IS BREAK CODE
E788 A6 00                  LDA A   X
E78A 84 03                  AND A   #$03        CONVERT CODE TO #
E78C 3F                     SWI
E78D 1D                     FCB     PBYTE       PRINT BREAK #
E78E 30                     TSX                 MOVE CURRENT SP TO IX
E78F 8D 0A                  BSR     OUTREG      PRINT REGISTERS
E791 7E E3 EE               JMP     DBGENT      GO ENTER DEBUGGER


E794 30           PREGSO    TSX                 MOVE CURRENT SP TO IX
E795 8D 04                  BSR     OUTREG      GO PRINT REGISTERS
E797 3B                     RTI


E798 30           RDUMP     TSX                 MOVE CURRENT SP TO IX
E799 08                     INX
E79A 08                     INX                 ADJUST FOR RETURN ADDR ON STACK


E79B 3F           OUTREG    SWI
E79C 16                     FCB     PCRLF
E79D 09                     DEX     FIX         SP VALUE
E79E FF A0 0F               STX     TEMP        SAVE SP VALUE
E7A1 08                     INX                 FIX IX TO POINT TO CC
E7A2 8D 18                  BSR     HEX2        PRINT CC
E7A4 8D 16                  BSR     HEX2        PRINT ACCB
E7A6 8D 14                  BSR     HEX2        PRINT ACCA
E7A8 8D 0D                  BSR     HEX4        PRINT IX
E7AA 8D 0B                  BSR     HEX4        PRINT PC
```

```
E7AC B6 A0 0F           LDA A  TEMP      PRINT SP
E7AF 3F                 SWI
E7B0 1D                 FCB    PBYTE
E7B1 B6 A0 10           LDA A  TEMP+1
E7B4 3F                 SWI
E7B5 1D                 FCB    PBYTE
E7B6 39                 RTS
E7B7 A6 00     HEX4     LDA A  X         GET A BYTE
E7B9 3F                 SWI
E7BA 1D                 FCB    PBYTE     PRINT IT
E7BB 08                 INX              MOVE POINTER
E7BC A6 00     HEX2     LDA A  X         GET A BYTE
E7BE 3F                 SWI
E7BF 1D                 FCB    PBYTE     PRINT IT
E7C0 08                 INX              MOVE POINTER
E7C1 3F                 SWI
E7C2 15                 FCB    PSPACE    PRINT A SPACE AFTER NUMBER
E7C3 39                 RTS


E7C4 BD E6 DE  BREAK    JSR    GETNUM    GET ADDRESS TO SET BREAKPOINT AT
E7C7 FF A0 0F           STX    TEMP      SAVE A COPY IN TEMP
E7CA 27 2B              BEQ    NOBRK     IF ADDRESS IS 0 NOT ALLOWED
E7CC 8D 2A              BSR    GET123    INPUT A 0,1,2, OR 3 OR QUIT
E7CE FE A0 0F           LDX    TEMP
E7D1 E6 00              LDA B  X         GET CODE AT BREAK ADDRESS
E7D3 37                 PSH B            STORE IN STACK
E7D4 E6 01              LDA B  1,X
E7D6 37                 PSH B
E7D7 16                 TAB
E7D8 CB 18              ADD B  #24       CONVERT TO BREAK CODE #
E7DA E7 01              STA B  1,X       INSERT
E7DC C6 3F              LDA B  #$3F      INSERT BREAKPOINT OPCODE
E7DE E7 00              STA B  X
E7E0 48                 ASL A            MULT BREAK # X 4
E7E1 48                 ASL A
E7E2 CE A0 24           LDX    #BRKTBL   ADD OFFSET TO BREAKTABLE
E7E5 3F                 SWI
E7E6 04                 FCB    ADDAX
E7E7 32                 PUL A  STORE     SAVED CODE IN BREAKTABLE
E7E8 A7 03              STA A  3,X
E7EA 32                 PUL A
E7EB A7 02              STA A  2,X
E7ED B6 A0 0F           LDA A  TEMP      STORE BREAK ADDR IN BREAKTABLE
E7F0 A7 00              STA A  X
E7F2 B6 A0 10           LDA A  TEMP+1
E7F5 A7 01              STA A  1,X
E7F7 39        NOBRK    RTS


E7F8 3F        GET123   SWI
E7F9 1E                 FCB    HEXIN
E7FA 29 0C              BVS    NOGET1
E7FC 5D                 TST B
E7FD 26 05              BNE    NOGET
E7FF 85 FC              BIT A  #%11111100 TEST FOR NOT 0,1,2,3
E801 26 01              BNE    NOGET
E803 39                 RTS
E804 86 3F     NOGET    LDA A  #'?
```

```
E806 3F                  SWI
E807 11                  FCB    OUTCHR
E808 31        NOGET1    INS    POP      1 RETURN ADDRESS OFF STACK
E809 31                  INS
E80A 39                  RTS             RETURN TO DEBUGGER MAINLINE


E80B 8D EB     KILBRK    BSR    GET123   INPUT A 0,1,2, OR 3 OR QUIT
E80D B7 A0 0F            STA A  TEMP
E810 48                  ASL A
E811 48                  ASL A           MULT X 4
E812 CE A0 24            LDX    #BRKTBL  ADD TO BREAK TABLE ADDRESS
E815 3F                  SWI
E816 04                  FCB    ADDAX
E817 86 04               LDA A  #4
E819 E6 03     KILL1     LDA B  3,X      CLEAR ENTRY &
E81B 6F 03               CLR    3,X      SAVE CONTENTS ON STACK
E81D 09                  DEX
E81E 37                  PSH B
E81F 4A                  DEC A
E820 26 F7               BNE    KILL1
E822 33                  PUL B
E823 32                  PUL A
E824 3F                  SWI
E825 03                  FCB    TBAX
E826 F6 A0 0F            LDA B  TEMP
E829 CB 18               ADD B  #24      CONVERT COPY OF # TO CODE
E82B E1 01               CMP B  1,X      CHECK FOR THAT # BREAK THERE
E82D 33                  PUL B
E82E 32                  PUL A
E82F 26 19               BNE    NOKILL
E831 E7 00               STA B  X        RESTORE SAVED CODE
E833 A7 01               STA A  1,X
E835 08                  INX
E836 08                  INX             MAKE IX = POSSIBLE RETURN ADDR
E837 3F                  SWI
E838 02                  FCB    TXBA     MOVE ADDRESS TO ACCUMULATORS
E839 30                  TSX
E83A A1 08               CMP A  8,X      COMPARE TO STACKED PC
E83C 26 0C               BNE    NOKILL
E83E E1 07               CMP B  7,X
E840 26 08               BNE    NOKILL
E842 80 02               SUB A  #2       MOVE PC BACK TO ADDR OF BREAK
E844 C2 00               SBC B  #0
E846 A7 08               STA A  8,X
E848 E7 07               STA B  7,X      STORE IN STACKED PC
E84A 39        NOKILL    RTS


E84B CE A0 24  PRTBRK    LDX    #BRKTBL  SET POINTER TO START
E84E 5F                  CLR B           SET COUNTER TO 0
E84F 3F        PRTBR1    SWI
E850 16                  FCB    PCRLF
E851 17                  TBA    COPY     BREAK # TO ACCA
E852 3F                  SWI
E853 1D                  FCB    PBYTE    PRINT BREAK #
E854 3F                  SWI
E855 15                  FCB    PSPACE
E856 BD E7 B7            JSR    HEX4     PRINT BREAK ADDRESS
```

```
E859 08                 INX
E85A 08                 INX                 MOVE PTR TO NEXT ENTRY IN TABLE
E85B 5C                 INC  B              INCREMENT COUNTER
E85C C1 04              CMP  B  #4          QUIT AT 4
E85E 26 EF              BNE     PRTBR1
E860 39                 RTS


               * WRITE A MIKBUG FORMAT TAPE
               * DATA IS FROM BEGIN1 TO END1
E861 BD E8 E7  MKWRIO   JSR     INITCT      SET UP LINE LENGTH COUNTER
E864 C6 3C              LDA  B  #60         WRITE 60 NULS TO TAPE AS LEADER
E866 4F        MKW1     CLR  A              ACCA GETS NUL VALUE
E867 3F        MKW11    SWI
E868 13                 FCB     TAPOUT
E869 5A                 DEC  B
E86A 26 FB              BNE     MKW11
E86C FE A0 02           LDX     BEGIN1      LOAD STARTING ADDRESS
E86F 3F                 SWI
E870 02                 FCB     TXBA
E871 FE A0 04           LDX     END1        LOAD END ADDRESS
E874 3F                 SWI                 SUBTRACT START ADDRESS
E875 07                 FCB     SUBBAX      RESULT IS # OF BYTES REMAINING
E876 08                 INX                 ADJUST FOR INCLUSIVE
E877 27 4A              BEQ     MKW7        IF RESULT = 0 THEN EXIT
E879 3F                 SWI
E87A 02                 FCB     TXBA
E87B 5D                 TST  B
E87C 26 04              BNE     MKW2        IF RESULT > 64 SET COUNT TO 64+3
E87E 81 40              CMP  A  #64
E880 23 02              BLS     MKW3
E882 86 40     MKW2     LDA  A  #64
E884 8B 03     MKW3     ADD  A  #3          ADD 3 FOR ADDRESS AND CHECKSUM
E886 B7 A0 0F           STA  A  TEMP        USE TEMP AS BYTE COUNTER
E889 86 53              LDA  A  #'S         WRITE OUT "S1"
E88B 3F                 SWI
E88C 13                 FCB     TAPOUT
E88D 86 31              LDA  A  #'1
E88F 3F                 SWI
E890 13                 FCB     TAPOUT
E891 7F A0 0A           CLR     SUBTOT      ZERO CHECKSUM ACCUMULATOR
E894 B6 A0 0F           LDA  A  TEMP        WRITE OUT BYTE COUNT
E897 8D 35              BSR     FRAME
E899 B6 A0 02           LDA  A  BEGIN1      WRITE ADDRESS TO TAPE
E89C 8D 30              BSR     FRAME
E89E B6 A0 03           LDA  A  BEGIN1+1
E8A1 8D 2B              BSR     FRAME
E8A3 FE A0 02           LDX     BEGIN1      GET POINTER TO DATA TO WRITE
E8A6 A6 00     MKW4     LDA  A  X           WRITE A BYTE OF DATA TO TAPE
E8A8 8D 24              BSR     FRAME
E8AA 26 FA              BNE     MKW4        CONTINUE TILL COUNT=0
E8AC FF A0 02           STX     BEGIN1      STORE POINTER
E8AF B6 A0 0A           LDA  A  SUBTOT      GET CHECKSUM
E8B2 43                 COM  A              INVERT
E8B3 8D 19              BSR     FRAME       WRITE OUT CHECKSUM
E8B5 8D 25              BSR     RCDMRK      PRINT "#" FOR EACH RECORD
E8B7 86 0D     MKW5     LDA  A  #CR         WRITE CR, LF TO TAPE
E8B9 3F                 SWI
E8BA 13                 FCB     TAPOUT
```

```
E8BB 86 0A              LDA A  #LF
E8BD 3F                 SWI
E8BE 13                 FCB    TAPOUT
E8BF C6 02              LDA B  #2        WRITE 2 NULS TO TAPE
E8C1 20 A3              BRA    MKW1
E8C3 86 53     MKW7     LDA A  #'S       WRITE OUT S9
E8C5 3F                 SWI
E8C6 13                 FCB    TAPOUT
E8C7 86 39              LDA A  #'9
E8C9 3F                 SWI
E8CA 13                 FCB    TAPOUT
E8CB 8D 01              BSR    FRAME
E8CD 3B                 RTI
E8CE 8D 1D     FRAME    BSR    MAKCHK    ADD BYTE TO CHECKSUM
E8D0 3F                 SWI
E8D1 0D                 FCB    BINHEX    CONVERT BYTE TO 2 HEX DIGITS
E8D2 3F                 SWI
E8D3 13                 FCB    TAPOUT    WRITE OUT MS DIGIT
E8D4 17                 TBA              MOVE LS DIGIT TO ACCA
E8D5 3F                 SWI
E8D6 13                 FCB    TAPOUT    WRITE OUT LS DIGIT
E8D7 08                 INX              MOVE PTR IF NEEDED; IF NOT NO HARM
E8D8 7A A0 0F           DEC    TEMP      DECREMENT BYTE COUNTER
E8DB 39                 RTS
               *
               * CHAR COUNT & CRLFS NEEDED IN CASE OUTPUT GOES TO A
               * HARD COPY DEVICE
E8DC 86 23     RCDMRK   LDA A  #'#       PRINT "#" FOR EACH RECORD
E8DE 3F                 SWI
E8DF 11                 FCB    OUTCHR
E8E0 7A A0 0B           DEC    SUBTOT+1  DECREMENT LENGTH OF LINE COUNTER
E8E3 26 07              BNE    RCDM9
E8E5 3F                 SWI
E8E6 16                 FCB    PCRLF     IF COUNTER=0 THEN PRINT CR-LF
E8E7 86 40     INITCT   LDA A  #64
E8E9 B7 A0 0B           STA A  SUBTOT+1  RESET COUNTER
E8EC 39        RCDM9    RTS
               *
E8ED 16        MAKCHK   TAB              COPY DATA
E8EE FB A0 0A           ADD B  SUBTOT    ADD CHECKSUM
E8F1 F7 A0 0A           STA B  SUBTOT    STORE UPDATED CHECKSUM
E8F4 39                 RTS
               *
               * SET ACIA FORMAT
E8F5 C6 03     SETFMT   LDA B  #3
E8F7 F7 80 00           STA B  CASCTL
E8FA B7 80 00           STA A  CASCTL
E8FD 39                 RTS
               *
               * MIKBUG TAPE READ ROUTINE
E8FE 8D E7     MKREA0   BSR    INITCT    INIT LINE LENGTH COUNTER
E900 86 15              LDA A  #$15
E902 8D F1              BSR    SETFMT    SET UP ACIA FOR 1 STOP BIT
E904 3F        MKR1     SWI
E905 12                 FCB    TAPEIN    GET A CHARACTER
E906 81 53     MKR2     CMP A  #'S       TEST FOR S
E908 26 FA              BNE    MKR1      WAIT FOR S
E90A 3F                 SWI
E90B 12                 FCB    TAPEIN
```

```
E90C 81 31                 CMP A  #'1        TEST FOR 1 (RECORD START)
E90E 27 0C                 BEQ    MKR3
E910 81 39                 CMP A  #'9        TEST FOR 9 (END OF FILE)
E912 26 F2                 BNE    MKR2       IF NOT 1 OR 9 TEST FOR S
E914 FF A0 14              STX    BEGIN2     STORE POINTER
E917 86 15                 LDA A  #$15       RESET ACIA TO 2 STOP BITS
E919 8D DA                 BSR    SETFMT
E91B 3B                    RTI               EXIT
E91C 7F A0 0A    MKR3      CLR    SUBTOT     ZERO CHECKSUM ACCUMULATOR
E91F 8D 3A                 BSR    IFRAME     INPUT 1ST BYTE: BYTE COUNT
E921 80 03                 SUB A  #3         ADJUST FOR COUNT & ADDRESS
E923 B7 A0 0F              STA A  TEMP       TEMP IS BYTE COUNTER
E926 8D 33                 BSR    IFRAME     INPUT 2 BYTES: ADDRESS
E928 B7 A0 14              STA A  BEGIN2     STORE ADDRESS IN BEGIN2
E92B 8D 2E                 BSR    IFRAME
E92D B7 A0 15              STA A  BEGIN2+1
E930 7D A0 10              TST    TEMP+1     TEST ADDRESS USE FLAG
E933 26 03                 BNE    MKR4       IF NOT 0 IGNORE ADDRESSES
E935 FE A0 14              LDX    BEGIN2     IF 0 LOAD PTR WITH TAPE ADDR
E938 8D 21       MKR4      BSR    IFRAME     GET A BYTE OF DATA
E93A A7 00                 STA A  X          STORE IT
E93C 08                    INX               MOVE STORAGE POINTER
E93D 7A A0 0F              DEC    TEMP       CHECK BYTE COUNT
E940 26 F6                 BNE    MKR4       CONTINUE TILL 0
E942 8D 17                 BSR    IFRAME     READ IN CHECKSUM
E944 73 A0 0A              COM    SUBTOT     CHECKSUM SHOULD BE $FF
E947 27 0E                 BEQ    MKR5
E949 86 3F                 LDA A  #'?        PRINT "?" IF RECORD HAS ERR
E94B 3F                    SWI
E94C 11                    FCB    OUTCHR
E94D B6 A0 14              LDA A  BEGIN2     PRINT ADDR OF RECORD ON ERR
E950 3F                    SWI
E951 1D                    FCB    PBYTE
E952 B6 A0 15              LDA A  BEGIN2+1
E955 3F                    SWI
E956 1D                    FCB    PBYTE
E957 8D 83       MKR5      BSR    RCDMRK     PRINT "#" FOR EACH RECORD
E959 20 A9                 BRA    MKR1       GO READ ANOTHER RECORD
E95B 3F          IFRAME    SWI
E95C 12                    FCB    TAPEIN     GET MS DIGIT
E95D 8D 0D                 BSR    IFR1       CONVERT TO BINARY
E95F 16                    TAB               MOVE TO ACCB HI BITS
E960 58                    ASL B
E961 58                    ASL B
E962 58                    ASL B
E963 58                    ASL B
E964 3F                    SWI
E965 12                    FCB    TAPEIN     GET LS DIGIT
E966 8D 04                 BSR    IFR1       CONVERT TO BINARY
E968 1B                    ABA               ADD IN LS DIGIT
E969 8D 82                 BSR    MAKCHK     ADD BYTE TO CHECKSUM
E96B 39                    RTS
E96C 80 30       IFR1      SUB A  #'0        REMOVE ASCII OFFSET
E96E 81 0A                 CMP A  #10        TEST FOR A-F
E970 2D 02                 BLT    IFR2
E972 80 07                 SUB A  #7         CONVERT A-F WITH EXTRA -7
E974 39          IFR2      RTS
```

```
                         * CARRIAGE RETURN CALL
E975 86 0D      PCRO      LDA A   #CR
E977 20 0C                BRA     OUTCHO
                         *
                         * CARRIAGE RETURN/LINE FEED CALL
E979 3F         PCRLFO    SWI
E97A 23                   FCB     PCR
                         *
                         * LINE FEED CALL
E97B 86 0A      PLFO      LDA A   #LF
E97D 20 06                BRA     OUTCHO
                         *
                         * SPACE CALL
E97F 86 20      PSPACO    LDA A   #SPACE
E981 20 02                BRA     OUTCHO
                         *
                         * FORM FEED CALL
E983 86 0C      PFFEEO    LDA A   #FF
                         *


                         * MAINLINE OF VIDEO DRIVER
E985 7D A0 39   OUTCHO    TST     WAITON
E988 27 10                BEQ     OUTCHZ
E98A 36                   PSH A
E98B 3F                   SWI
E98C 26                   FCB     INKEY
E98D 27 0A                BEQ     OUTCHX
E98F 81 13                CMP A   #$13        TEST FOR CTL-S
E991 26 06                BNE     OUTCHX
E993 3F         OUTCHY    SWI
E994 10                   FCB     INCHAR
E995 81 13                CMP A   #$13
E997 26 FA                BNE     OUTCHY
E999 32         OUTCHX    PUL A
E99A 7D A0 1C   OUTCHZ    TST     ESCSTR      ARE WE PROCESSING ESC STRING?
E99D 27 05                BEQ     OUTCH1
E99F FE A0 3B             LDX     ESCVEC      GO TO ACTIVE ESCAPE ROUTINE
E9A2 6E 00      OUTCHJ    JMP     X
E9A4 81 1B      OUTCH1    CMP A   #ESC        TEST FOR ESCAPE
E9A6 27 14                BEQ     CCHAR1      ESC IS PROCESSED BY ESC ROUTINE
E9A8 7D A0 36             TST     OUTPTR      CHECK FOR PRINTER ENABLED
E9AB 27 04                BEQ     OUTCH2
E9AD 3F                   SWI
E9AE 27                   FCB     PRINT
E9AF 2B 12                BMI     OUTCH3
E9B1 85 E0      OUTCH2    BIT A   #$E0        TEST FOR CONTROL CHARACTERS
E9B3 27 07                BEQ     CCHAR1      GO PROCESS CONTROL CHAR
E9B5 81 7F                CMP A   #$7F        TEST FOR DELETE
E9B7 27 0A                BEQ     OUTCH3
E9B9 7E EB 80             JMP     PCHAR       GO PROCESS PRINTING CHARACTER
E9BC CE E9 C4   CCHAR1    LDX     #VDTABL
E9BF 3F                   SWI
E9C0 28                   FCB     TBSRCH
E9C1 27 DF                BEQ     OUTCHJ
                         *
E9C3 3B         OUTCH3    RTI
                         *
E9C4 07         VDTABL    FCB     BELL
```

```
E9C5 EA 47                    FDB    DOBELL
E9C7 08                       FCB    BS
E9C8 EA 30                    FDB    BSPACE
E9CA 0C                       FCB    FF
E9CB EA 07                    FDB    FFEED
E9CD 0A                       FCB    LF
E9CE EB DD                    FDB    LFEED
E9D0 0D                       FCB    CR
E9D1 E9 D7                    FDB    CRETRN
E9D3 1B                       FCB    ESC
E9D4 EA 62                    FDB    ESCAPE
E9D6 00                       FCB    0
                     *
                     * CARRIAGE RETURN ROUTINE
E9D7 7F F9 02        CRETRN   CLR    XCRSR
E9DA F6 A0 1A                 LDA B  WRAP
E9DD 26 0F                    BNE    CR2
E9DF 7D F9 03                 TST    YCRSR
E9E2 26 07                    BNE    CR1
E9E4 86 17                    LDA A  #HEIGHT-1
E9E6 B7 F9 03                 STA A  YCRSR
E9E9 20 03                    BRA    CR2
E9EB 7A F9 03        CR1      DEC    YCRSR
E9EE 86 50           CR2      LDA A  #LINE
E9F0 B7 A0 1A                 STA A  WRAP
E9F3 10                       SBA
E9F4 27 10                    BEQ    CR4
E9F6 FE A0 18                 LDX    CURSOR
E9F9 8C F0 00                 CPX    #SCREEN
E9FC 26 03                    BNE    CR3
E9FE CE F7 80                 LDX    #ENDSCN
EA01 3F              CR3      SWI
EA02 06                       FCB    SUBAX
EA03 FF A0 18                 STX    CURSOR
EA06 3B              CR4      RTI
                     *
                     * FORM FEED ROUTINE
EA07 7F F9 03        FFEED    CLR    YCRSR      HOME CURSOR
EA0A 7F F9 02                 CLR    XCRSR
EA0D 86 09                    LDA A  #VDBLNK
EA0F 3F                       SWI
EA10 29                       FCB    ESCSND
EA11 CE F0 00                 LDX    #SCREEN
EA14 FF A0 18                 STX    CURSOR
EA17 86 17                    LDA A  #HEIGHT-1
EA19 B7 F9 01                 STA A  SCROLL
EA1C 86 50                    LDA A  #LINE      SET UP WRAP AROUND COUNTER
EA1E B7 A0 1A                 STA A  WRAP
EA21 86 20                    LDA A  #SPACE     FILL SCREEN WITH SPACES
EA23 A7 00           FF1      STA A  X
EA25 08                       INX
EA26 8C F7 80                 CPX    #ENDSCN
EA29 26 F8                    BNE    FF1
EA2B 86 0A                    LDA A  #VDNORM
EA2D 3F                       SWI
EA2E 29                       FCB    ESCSND
EA2F 3B                       RTI
                     *
EA30 7D F9 02        BSPACE   TST    XCRSR
```

```
EA33 27 11             BEQ      BS1
EA35 7A F9 02          DEC      XCRSR
EA38 7C A0 1A          INC      WRAP
EA3B FE A0 18          LDX      CURSOR
EA3E 09                DEX
EA3F 86 20             LDA A    #SPACE
EA41 A7 00             STA A    X
EA43 FF A0 18          STX      CURSOR
EA46 3B        BS1     RTI
               *
EA47 FE A0 3F  DOBELL  LDX      DURATN
EA4A B6 F9 00          LDA A    CONTRL
EA4D 84 7F             AND A    #$7F
EA4F 88 80     BELL1   EOR A    #$80
EA51 B7 F9 00          STA A    CONTRL
EA54 F6 A0 41          LDA B    PERIOD
EA57 09        BELL2   DEX
EA58 26 03             BNE      BELL3
EA5A 7E EA F4          JMP      STORBT
EA5D 5A        BELL3   DEC B
EA5E 26 F7             BNE      BELL2
EA60 20 ED             BRA      BELL1
               *
EA62 CE EA 6A  ESCAPE  LDX      #ESCPRC
EA65 7C A0 1C          INC      ESCSTR
EA68 20 52             BRA      POS1
               *
EA6A 4A        ESCPRC  DEC A
EA6B 81 14             CMP A    #20
EA6D 22 6C             BHI      ESCOF1
EA6F CE EA 79          LDX      #ESCTBL
EA72 48                ASL A
EA73 3F                SWI
EA74 04                FCB      ADDAX
EA75 EE 00             LDX      X
EA77 6E 00             JMP      X


               *
EA79 EA A3     ESCTBL  FDB      POSXO
EA7B EA B9             FDB      POSYO
EA7D EA EF             FDB      MEMOFO
EA7F EA FD             FDB      MEMONO
EA81 EB 04             FDB      SLCVIO
EA83 EB 08             FDB      SLCFNO
EA85 EB 0C             FDB      SETMOO
EA87 EB 10             FDB      SETM1O
EA89 EB 14             FDB      VDBLNO
EA8B EB 18             FDB      VDNORO
EA8D EB 1C             FDB      CBOFFO
EA8F EB 20             FDB      CBONO
EA91 EB 24             FDB      CBSTDO
EA93 EB 28             FDB      CBFLSO
EA95 EB 2C             FDB      OUTSCO
EA97 EB 2F             FDB      OUTPRO
EA99 EB 36             FDB      OUTBTO
EA9B EB 3A             FDB      SLOTOO
EA9D EB 3D             FDB      SLOT1O
EA9F EA DD             FDB      PRGMDO
```

```
EAA1 EA E2                FDB    TRMODO
                  *
EAA3 CE EA A8    POSX0    LDX    #POSX1
EAA6 20 14                BRA    POS1
                  *
EAA8 81 4F       POSX1    CMP A  #LINE-1
EAAA 23 02                BLS    POSX2
EAAC 86 4F                LDA A  #LINE-1
EAAE B7 F9 02    POSX2    STA A  XCRSR
EAB1 80 50                SUB A  #LINE
EAB3 40                   NEG A
EAB4 B7 A0 1A             STA A  WRAP
EAB7 20 10                BRA    MAKCSR
                  *
EAB9 CE EA C0    POSY0    LDX    #POSY1
EABC FF A0 3B    POS1     STX    ESCVEC
EABF 3B                   RTI
                  *
EAC0 81 17       POSY1    CMP A  #HEIGHT-1
EAC2 23 02                BLS    POSY2
EAC4 86 17                LDA A  #HEIGHT-1
EAC6 B7 F9 03    POSY2    STA A  YCRSR
                  *
EAC9 B6 F9 03    MAKCSR   LDA A  YCRSR
EACC C6 50                LDA B  #LINE
EACE 3F                   SWI
EACF 08                   FCB    MULTAB
EAD0 BB F9 02             ADD A  XCRSR
EAD3 C9 F0                ADC B  #SCREEN/256
EAD5 B7 A0 19             STA A  CURSOR+1
EAD8 F7 A0 18             STA B  CURSOR
EADB 20 1C       ESCOF1   BRA    ESCOFF
                  *
EADD CE EB 44    PRGMOO   LDX    #PRG1
EAE0 20 DA                BRA    POS1
                  *
EAE2 CE EA E7    TRMODO   LDX    #TRM1
EAE5 20 D5                BRA    POS1
                  *
EAE7 B7 A0 1C    TRM1     STA A  ESCSTR
EAEA CE EB 63             LDX    #TRM2
EAED 20 CD                BRA    POS1
                  *
EAEF 86 7E       MEMOFO   LDA A  #%01111110
EAF1 B4 F9 00    BITOFF   AND A  CONTRL
EAF4 84 7F       STORBT   AND A  #$7F
EAF6 B7 F9 00             STA A  CONTRL
EAF9 7F A0 1C    ESCOFF   CLR    ESCSTR
EAFC 3B                   RTI
                  *
EAFD 86 01       MEMONO   LDA A  #%00000001
EAFF BA F9 00    BITON    ORA A  CONTRL
EB02 20 F0                BRA    STORBT
                  *
EB04 86 7D       SLCVIO   LDA A  #%01111101
EB06 20 E9                BRA    BITOFF
                  *
EB08 86 02       SLCFNO   LDA A  #%00000010
EB0A 20 F3                BRA    BITON
```

```
                    *
EB0C 86 77          SETM00  LDA A   #%01110111
EB0E 20 E1                  BRA     BITOFF
                    *
EB10 86 08          SETM10  LDA A   #%00001000
EB12 20 EB                  BRA     BITON
                    *
EB14 86 6F          VDBLN0  LDA A   #%01101111
EB16 20 D9                  BRA     BITOFF
                    *
EB18 86 10          VDNOR0  LDA A   #%00010000
EB1A 20 E3                  BRA     BITON
                    *
EB1C 86 5F          CBOFF0  LDA A   #%01011111
EB1E 20 D1                  BRA     BITOFF
                    *
EB20 86 20          CBON0   LDA A   #%00100000
EB22 20 DB                  BRA     BITON
                    *
EB24 86 BF          CBSTD0  LDA A   #%10111111
EB26 20 C9                  BRA     BITOFF
                    *
EB28 86 40          CBFLS0  LDA A   #%01000000
EB2A 20 D3                  BRA     BITON
                    *
EB2C 4F             OUTSC0  CLR A
EB2D 20 02                  BRA     OUTP1
                    *
EB2F 86 FF          OUTPR0  LDA A   #$FF
EB31 B7 A0 36       OUTP1   STA A   OUTPTR
EB34 20 C3                  BRA     ESCOFF
                    *
EB36 86 01          OUTBT0  LDA A   #$01
EB38 20 F7                  BRA     OUTP1
                    *
EB3A 4F             SLOT00  CLR A
EB3B 20 02                  BRA     SL1
                    *
EB3D 86 80          SLOT10  LDA A   #$80
EB3F B7 A0 1B       SL1     STA A   CHRSLT
EB42 20 B5                  BRA     ESCOFF
                    *
EB44 B7 A0 1C       PRG1    STA A   ESCSTR
EB47 CE EB 56               LDX     #PRG2
EB4A FF A0 3B               STX     ESCVEC
EB4D 86 04                  LDA A   #%00000100
EB4F BA F9 00               ORA A   CONTRL
EB52 B7 F9 00               STA A   CONTRL
EB55 3B                     RTI
                    *
EB56 B7 F9 01       PRG2    STA A   MODE
EB59 7A A0 1C               DEC     ESCSTR
EB5C 27 01                  BEQ     PRG3
EB5E 3B             TRM3    RTI
                    *
EB5F 86 7B          PRG3    LDA A   #%01111011
EB61 20 8E                  BRA     BITOFF
                    *
EB63 7A A0 1C       TRM2    DEC     ESCSTR
```

```
EB66 7D A0 36            TST       OUTPTR
EB69 27 15               BEQ       PCHAR
EB6B 36                  PSH  A
EB6C 81 7F               CMP  A    #$7F
EB6E 27 06               BEQ       TRM25
EB70 85 E0               BIT  A    #$E0
EB72 2B 02               BMI       TRM25
EB74 26 02               BNE       TRM4
EB76 86 2E      TRM25    LDA  A    #'.
EB78 3F         TRM4     SWI
EB79 27                  FCB       PRINT
EB7A 32                  PUL  A
EB7B 7D A0 36            TST       OUTPTR
EB7E 2B DE               BMI       TRM3
                *
EB80 FE A0 18   PCHAR    LDX       CURSOR
EB83 BA A0 1B            ORA  A    CHRSLT
EB86 A7 00               STA  A    X
EB88 08                  INX
EB89 8C F7 80            CPX       #ENDSCN
EB8C 26 03               BNE       PCHAR1
EB8E CE F0 00            LDX       #SCREEN
EB91 FF A0 18   PCHAR1   STX       CURSOR
EB94 7C F9 02            INC       XCRSR
EB97 7A A0 1A            DEC       WRAP
EB9A 27 08               BEQ       PCHAR3
EB9C 2A 05               BPL       PCHAR2
EB9E 86 4F               LDA  A    #LINE-1
EBA0 B7 A0 1A            STA  A    WRAP
EBA3 3B         PCHAR2   RTI
EBA4 7F F9 02   PCHAR3   CLR       XCRSR
                *
                * UPDATE YCRSR
EBA7 F6 F9 03   UY       LDA  B    YCRSR
EBAA 4F                  CLR  A
EBAB C1 17               CMP  B    #HEIGHT-1
EBAD 27 02               BEQ       UY1
EBAF 17                  TBA
EBB0 4C                  INC  A
EBB1 B7 F9 03   UY1      STA  A    YCRSR
                *
                * UPDATE SCROLL
EBB4 7D F9 00   US0      TST       CONTRL
EBB7 2A FB               BPL       US0
EBB9 F1 F9 01            CMP  B    SCROLL
EBBC 26 1E               BNE       US99
EBBE C1 17               CMP  B    #HEIGHT-1
EBC0 27 05               BEQ       US1
EBC2 7C F9 01            INC       SCROLL
EBC5 20 03               BRA       US3
EBC7 7F F9 01   US1      CLR       SCROLL
EBCA FE A0 18   US3      LDX       CURSOR
EBCD B6 F9 02            LDA  A    XCRSR
EBD0 3F                  SWI
EBD1 06                  FCB       SUBAX
EBD2 86 20               LDA  A    #SPACE
EBD4 C6 50               LDA  B    #LINE
EBD6 A7 00      US9      STA  A    X
EBD8 08                  INX
```

```
EBD9 5A                 DEC B
EBDA 26 FA              BNE    US9
EBDC 3B        US99     RTI
               *
EBDD FE A0 18  LFEED    LDX    CURSOR
EBE0 86 50              LDA A  #LINE
EBE2 F6 F9 03           LDA B  YCRSR
EBE5 C1 17              CMP B  #HEIGHT-1
EBE7 26 06              BNE    UC1
EBE9 CE F0 00           LDX    #SCREEN
EBEC B6 F9 02           LDA A  XCRSR
EBEF 3F        UC1      SWI
EBF0 04                 FCB    ADDAX
EBF1 FF A0 18           STX    CURSOR
EBF4 20 B1              BRA    UY
               *
EBF6 3F        PSTRN1   SWI
EBF7 11                 FCB    OUTCHR     PRINT CHARACTER
EBF8 08                 INX               MOVE POINTER
               * ENTER HERE
EBF9 A6 00     PSTRN0   LDA A  X          GET CHARACTER FROM STRING
EBFB 81 04              CMP A  #4         TEST FOR CARRIAGE RETURN
EBFD 26 F7              BNE    PSTRN1
EBFF 3B                 RTI
EC00           AAAAAA   EQU    *
                        END

NO ERROR(S) DETECTED


 SYMBOL TABLE:
AAAAAA EC00   ADDAX  0004   ADDAX0 E266   ADDBA0 E267   ADDBAX 0005
ARITHM E5AE   ASC1   E4A6   ASC2   E4B3   ASC3   E4BB   ASCENT E4A1
ASSIGN E498   BCDHEX E317   BEGIN1 A002   BEGIN2 A014   BELL   0007
BELL1  EA4F   BELL2  EA57   BELL3  EA5D   BINHE0 E309   BINHEX 000D
BITOFF EAF1   BITON  EAFF   BLOCK  00FF   BREAK  E7C4   BREAK0 0018
BREAK1 0019   BREAK2 001A   BREAK3 001B   BRKPRT E77C   BRKTBL A024
BS     0008   BS1    EA46   BSPACE EA30   BYTECT A00B   CASCTL 8000
CASDAT 8001   CBFLS0 EB28   CBFLSH 000E   CBOFF  000B   CBOFF0 EB1C
CBON   000C   CBON0  EB20   CBSTD0 EB24   CBSTDY 000D   CCHAR1 E9BC
CHARB  E552   CHECK1 E5D7   CHECK2 E5E1   CHECK3 E5E6   CHKSUM E5CE
CHRSLT A01B   CKSM   A00A   CONTRL F900   CR     000D   CR1    E9EB
CR2    E9EE   CR3    EA01   CR4    EA06   CRETRN E9D7   CURSOR A018
DBGENT E3EE   DEB0   E400   DEB1   E420   DEBUG  E403   DEBUGR 001F
DO2    E4BD   DO20   E4C4   DO21   E4C9   DOBELL EA47   DOMINS E48E
DOP1   E48A   DOP2   E48D   DOPLUS E486   DUMP   E5F1   DUMP1  E5F4
DUMP2  E5FD   DUMP3  E61B   DUPLEX A038   DURATN A03F   END1   A004
END2   A016   ENDLIN A01D   ENDMOV E316   ENDSCN F780   ENDTBL E482
ENTER  E482   EOT    0004   ESC    001B   ESCAPE EA62   ESCOF1 EADB
ESCOFF EAF9   ESCPRC EA6A   ESCSND 0029   ESCSTR A01C   ESCTBL EA79
ESCVEC A03B   FF     000C   FF1    EA23   FFEED  EA07   FIND05 E749
FIND10 E74C   FIND20 E76C   FIND30 E774   FINDXT E730   FMTDMP E62E
FORM1  E631   FORM3  E665   FRAME  E8CE   GET123 E7F8   GETADS E6CF
GETB1  E731   GETBYT E724   GETN1  E6E4   GETNUM E6DE   GOTO   E675
GOTO3  E687   GOTO4  E68E   GOTO9  E691   HEIGHT 0018   HEX2   E7BC
HEX4   E7B7   HEXBI0 E2D8   HEXBI1 E2DC   HEXBI2 E2F0   HEXBI4 E304
HEXBI5 E314   HEXBIN 000C   HEXIN  001E   HXLOCT E735   IFR1   E96C
IFR2   E974   IFRAME E95B   INCAPS 002A   INCHAR 0010   INECHO 0025
INITC1 E54D   INITCS E542   INITCT E8E7   INITD1 E504   INITDX E4F9
```

INITL2	E4DC	INITL9	E4D2	INITLZ	E4D3	INITNL	E509	INITO1	E520
INITOP	E510	INITP1	E530	INITP2	E535	INITPF	E525	INITPX	E533
INITR1	E563	INITR2	E574	INITR3	E576	INITR4	E583	INITR5	E59E
INITRP	E558	INITSP	E536	INITUN	E397	INKEY	0026	INTTBL	E4E0
IOBLK	8000	IRQENT	E000	IRQVEC	A000	JSRTO	E693	JTABL1	E430
JTABLE	E445	KBDCTL	8011	KBDDAT	8010	KILBRK	E80B	KILL1	E819
LF	000A	LFEED	EBDD	LINBUF	A01F	LINE	0050	LINEIN	0020
LOAD	E697	LOAD1	E6A0	LOWLFT	F730	MAKBYT	E719	MAKCHK	E8ED
MAKCSR	EAC9	MBTEMP	A00E	MCONT	A011	MEMOF0	EAEF	MEMOFF	0003
MEMON	0004	MEMON0	EAFD	MEMORY	E6A6	MEMPTR	A021	MKR1	E904
MKR2	E906	MKR3	E91C	MKR4	E938	MKR5	E957	MKREA0	E8FE
MKREAD	0022	MKW1	E866	MKW11	E867	MKW2	E882	MKW3	E884
MKW4	E8A6	MKW5	E8B7	MKW7	E8C3	MKWRI0	E861	MKWRIT	0021
MMODE	A023	MMOFF	E494	MODE	F901	MOVE1	E32F	MOVE2	E339
MOVER	000E	MOVER0	E320	MULTA0	E280	MULTA1	E288	MULTA2	E28C
MULTAB	0008	MYCODE	E19B	NMIENT	E005	NMIVEC	A006	NMOVE	E33B
NMOVE1	E33E	NOADS	E6E2	NOBRK	E7F7	NOECHO	E428	NOGET	E804
NOGET1	E808	NOKILL	E84A	NULCNT	A037	OPEN	E316	OUTBT0	EB36
OUTBTH	0011	OUTCH0	E985	OUTCH1	E9A4	OUTCH2	E9B1	OUTCH3	E9C3
OUTCHJ	E9A2	OUTCHR	0011	OUTCHX	E999	OUTCHY	E993	OUTCHZ	E99A
OUTP1	EB31	OUTPR0	EB2F	OUTPRT	0010	OUTPTR	A036	OUTREG	E79B
OUTSC0	EB2C	OUTSCR	000F	PBYTE	001D	PBYTSP	E5C9	PCHAR	EB80
PCHAR1	EB91	PCHAR2	EBA3	PCHAR3	EBA4	PCLOCN	A048	PCR	0023
PCR0	E975	PCRLF	0016	PCRLF0	E979	PERIOD	A041	PFFEE0	E983
PFFEED	0017	PLF	0024	PLF0	E97B	POS1	EABC	POSX	0001
POSX0	EAA3	POSX1	EAA8	POSX2	EAAE	POSY	0002	POSY0	EAB9
POSY1	EAC0	POSY2	EAC6	PPCTL	8013	PPDATA	8012	PRBA	E5C4
PREGS	001C	PREGS0	E794	PRFLAG	A03E	PRG1	EB44	PRG2	EB56
PRG3	EB5F	PRGMO0	EADD	PRGMOD	0014	PRINT	0027	PRTBR1	E84F
PRTBRK	E84B	PRX	E5C2	PSHAL0	E295	PSHAL1	E29D	PSHAL2	E2A8
PSHALL	0009	PSP	E5CB	PSPAC0	E97F	PSPACE	0015	PSTRN0	EBF9
PSTRN1	EBF6	PSTRNG	0014	PULAL0	E2B1	PULAL1	E2B4	PULAL2	E2BE
PULALL	000A	RCDM9	E8EC	RCDMRK	E8DC	RDUMP	E798	RESET	000F
RESET0	E397	RESET1	E39A	RESET2	E3A6	RMOVE	E357	RMOVE1	E36B
RPRT	E5A3	SAVE	E6AF	SCREEN	F000	SCROLL	F901	SCRPAD	A000
SETFMT	E8F5	SETM0	0007	SETM00	EB0C	SETM1	0008	SETM10	EB10
SL1`	EB3F	SLASH	002F	SLCFN0	EB08	SLCFNT	0006	SLCVI0	EB04
SLCVID	0005	SLOT0	0012	SLOT00	EB3A	SLOT1	0013	SLOT10	EB3D
SP	A008	SPACE	0020	SPCTL	800C	SPDATA	800D	STAAL0	E2CC
STAAL1	E2CF	STAALL	000B	STAR	002A	STORBT	EAF4	STXENT	E26D
SUBAX	0006	SUBAX0	E272	SUBBA0	E273	SUBBAX	0007	SUBTOT	A00A
SWI0	E1D8	SWI1	E1F4	SWIENT	E191	SWIFLG	A03D	SWITBL	E1F7
TAPEIN	0012	TAPOUT	0013	TBAX	0003	TBAX0	E26C	TBSRCH	0028
TEMP	A00F	TEST1	E6EA	TEST2	E6EE	TEST3	E6FC	TEST5	E70E
TEST6	E716	TRM1	EAE7	TRM2	EB63	TRM25	EB76	TRM3	EB5E
TRM4	EB78	TRMOD0	EAE2	TRMODE	0015	TSTMEM	E6E5	TW	A00F
TXBA	0002	TXBA0	E25B	UC1	EBEF	UPCASE	A03A	US0	EBB4
US1	EBC7	US3	EBCA	US9	EBD6	US99	EBDC	USRCOD	E196
USRVEC	A034	UY	EBA7	UY1	EBB1	VDBLN0	EB14	VDBLNK	0009
VDINIT	E384	VDNOR0	EB18	VDNORM	000A	VDTABL	E9C4	WAITON	A039
WRAP	A01A	XBA	0000	XBA0	E260	XBAX	0001	XBAX0	E253
XCRSR	F902	XFER	E6B4	XHI	A00C	XLO	A00D	XTEMP	A012
YCRSR	F903	ZAP1	E6C6	ZAPMEM	E6BE				

```
                 ***********************************
                 *                                 *
                 *  CODE EXTENSION TO GMXBUG TO    *
                 *  PROVIDE MIKBUG COMPATIBILITY   *
                 *                                 *
                 *  BY Richard Rostrom  4/27/79    *
                 *                                 *
                 * COPYRIGHT (C) Gimix, Inc. 1979  *
                 *                                 *
                 ***********************************


                 * SYSTEM SYMBOL DEFINITION
                 *
                 * SYMBOLIC EQUATIONS
                 * ASCII CHARS
0004             EOT     EQU    $04
0007             BELL    EQU    $07
0008             BS      EQU    $08
000C             FF      EQU    $0C
000A             LF      EQU    $0A
000D             CR      EQU    $0D
001B             ESC     EQU    $1B
0020             SPACE   EQU    $20
002A             STAR    EQU    $2A        ASSEMBLER WILL NOT PROCESS
002F             SLASH   EQU    $2F        " #'* " OR " #'/ "
00FF             BLOCK   EQU    $FF


                 * VIDEO SCREEN DEFINITION
F000             SCREEN  EQU    $F000
0050             LINE    EQU    80         80 CHARACTERS/LINE
0018             HEIGHT  EQU    24
                 *
F780             ENDSCN  EQU    HEIGHT*LINE+SCREEN
F730             LOWLFT  EQU    ENDSCN-LINE
                 *
                 * VIDEO BOARD CONTROL REGISTERS
F900             VCNTRL  EQU    $F900
F901             MODE    EQU    $F901
F901             SCROLL  EQU    $F901
F902             XCRSR   EQU    $F902
F903             YCRSR   EQU    $F903


                 * I/O PORT ADDRESSES
8000             IOBLK   EQU    $8000
                 * KEYBOARD PIA IS PORT #4
8010             KBDDAT  EQU    $8010
8011             KBDCTL  EQU    $8011
                 * PARALLEL PRINTER PIA IS B SIDE PORT #4
8012             PPDATA  EQU    $8012
8013             PPCTL   EQU    $8013
                 * CASSETTE ACIA IS PORT #0
8000             CASCTL  EQU    $8000
8001             CASDAT  EQU    $8001
                 * SERIAL PRINTER ACIA IS PORT #3
800C             SPCTL   EQU    $800C
```

```
800D            SPDATA  EQU    $800D


                * SYSTEM CALL CODE ASSIGNMENTS
                * REGISTER AND ARITHMETIC FUNCTIONS
0000            XBA     EQU    0
0001            XBAX    EQU    1
0002            TXBA    EQU    2
0003            TBAX    EQU    3
0004            ADDAX   EQU    4
0005            ADDBAX  EQU    5
0006            SUBAX   EQU    6
0007            SUBBAX  EQU    7
0008            MULTAB  EQU    8
0009            PSHALL  EQU    9
000A            PULALL  EQU    10
000B            STAALL  EQU    11
                * MISCELLANEOUS
000C            HEXBIN  EQU    12
000D            BINHEX  EQU    13
000E            MOVER   EQU    14
000F            RESET   EQU    15
                * I/O FUNCTIONS
0010            INCHAR  EQU    16
0011            OUTCHR  EQU    17
0012            TAPEIN  EQU    18
0013            TAPOUT  EQU    19
0014            PSTRNG  EQU    20
0015            PSPACE  EQU    21
0016            PCRLF   EQU    22
0017            PFFEED  EQU    23
                * DEBUGGING FUNCTIONS
0018            BREAK0  EQU    24
0019            BREAK1  EQU    25
001A            BREAK2  EQU    26
001B            BREAK3  EQU    27
001C            PREGS   EQU    28
001D            PBYTE   EQU    29
001E            HEXIN   EQU    30
001F            DEBUGR  EQU    31
                * MISCELLANEOUS FUNCTIONS
0020            LINEIN  EQU    32
0021            MKWRIT  EQU    33
0022            MKREAD  EQU    34
0023            PCR     EQU    35
0024            PLF     EQU    36
0025            INECHO  EQU    37
0026            INKEY   EQU    38
0027            PRINT   EQU    39
0028            TBSRCH  EQU    40
0029            ESCSND  EQU    41
002A            INCAPS  EQU    42
                *
                * 43-45 TO BE FILLED IN LATER


                * CODES FOR ESCAPE FUNCTIONS
000B            CBOFF   EQU    11
000C            CBON    EQU    12
```

```
                * MONITOR MEMORY USAGE
                *
A000                    ORG     $A000
                *
A000            SCRPAD  EQU     *
                *
                * INTERRUPT REQUEST VECTOR
A000            IRQVEC  RMB     2
                *
                * GENERAL PURPOSE
A002            BEGIN1  RMB     2
A004            END1    RMB     2
                *
                * NON-MASKABLE INTERRUPT VECTOR
A006            NMIVEC  RMB     2
                *
                * MIKBUG EQUIVALENCE
A008            SP      RMB     2
                *
A00A            SUBTOT  EQU     *
                *
A00A            CKSM    RMB     1
                *
A00B            BYTECT  RMB     1
A00C            XHI     RMB     1
A00D            XLO     RMB     1
A00E            MBTEMP  RMB     1
                *
A00F            TEMP    EQU     *
                *
A00F            TW      RMB     2
A011            MCONT   RMB     1
A012            XTEMP   RMB     2
                *
                * GENERAL PURPOSE
A014            BEGIN2  RMB     2
A016            END2    RMB     2
                *
                * VIDEO DRIVER
A018            CURSOR  RMB     2
A01A            WRAP    RMB     1
A01B            CHRSLT  RMB     1
A01C            ESCSTR  RMB     1
                *
                * LINE INPUT
A01D            ENDLIN  RMB     2
A01F            LINBUF  RMB     2
                *
                * DEBUGGER STORAGE
A021            MEMPTR  RMB     2
A023            MMODE   RMB     1
                *
                * BREAKPOINT TABLE
A024            BRKTBL  RMB     16
                *
                * USER CODE VECTOR FOR SOFTWARE INTERRUPT
A034            USRVEC  RMB     2
                * OUTPUT DEVICE FLAG: $00=SCREEN ONLY, $01=BOTH,
```

```
                        * $FF=PRINTER ONLY
 A036                   OUTPTR   RMB     1
                        * # OF NULLS TO FOLLOW CR FOR PRINTER DRIVER
 A037                   NULCNT   RMB     1
                        * HALF/FULL DUPLEX FLAG  (1/0)
 A038                   DUPLEX   RMB     1
                        * CTL-S SWITCH: IF 0, CTL-S IS DISABLED
 A039                   WAITON   RMB     1
                        * FORCE UPPER CASE ALPAS FLAG
 A03A                   UPCASE   RMB     1
                        * VECTOR TO ACTIVE ESCAPE SEQUENCE ROUTINE
 A03B                   ESCVEC   RMB     2
                        * SWI CONTROL
 A03D                   SWIFLG   RMB     1
                        * PARALLEL/SERIAL PRINTER DRIVER (0/1)
 A03E                   PRFLAG   RMB     1
                        * BELL ROUTINE PARAMETERS
 A03F                   DURATN   RMB     2
 A041                   PERIOD   RMB     1


 A048                   PCLOCN   EQU     $A048


 E000                            ORG     $E000


                        * IRQ PROCESSOR
 E000 FE A0 00          IO       LDX     IRQVEC
 E003 6E 00                      JMP     X


                        * NMI PROCESSOR
 E005 FE A0 06          POWDWN   LDX     NMIVEC
 E008 6E 00                      JMP     X


                        * TAPE LOAD ROUTINE
 E00A 7F A0 10          LOAD     CLR     TEMP+1
 E00D 74 A0 3D                   LSR     SWIFLG
 E010 3F                         SWI
 E011 22                         FCB     MKREAD
 E012 20 3A                      BRA     L1


                        * LINEIN
                        * LINE INPUT ROUTINE
 E014 FE A0 1F          LINEIO   LDX     LINBUF      INITIALIZE POINTERS
 E017 C6 50                      LDA B   #LINE       SET LIMIT ON STRING LENGTH
 E019 3F                LINE1    SWI
 E01A 10                         FCB     INCHAR      INPUT A CHAR
 E01B 81 1B                      CMP A   #ESC        TEST FOR ESC
 E01D 27 0A                      BEQ     LINE15      ON ESC EXIT WITH ESC IN ACCA
 E01F 81 0D                      CMP A   #CR         TEST FOR CARRIAGE RETURN
 E021 26 0F                      BNE     LINE20
 E023 A7 00                      STA A   X           STORE CR IN BUFFER AS TERMINATOR
 E025 08                         INX                 BUMP POINTER
 E026 FF A0 1D                   STX     ENDLIN
 E029 30                LINE15   TSX
```

```
E02A A7 02              STA A  2,X       STORE TERMINATOR IN STACKED ACCA
E02C 50                 NEG B
E02D CB 50              ADD B  #LINE
E02F E7 01              STA B  1,X       STORE CHAR COUNT IN STACKED ACCB
E031 3B                 RTI
E032 81 08     LINE20   CMP A  #BS       TEST FOR BACK SPACE
E034 26 4F              BNE    LINE30
               * BACKSPACE ROUTINE
E036 BC A0 1F           CPX    LINBUF    CHECK FOR PTR AT START OF LINE
E039 27 DE              BEQ    LINE1
E03B 09                 DEX              MOVE POINTER BACK 1
E03C 5C                 INC B
E03D 3F                 SWI
E03E 11                 FCB    OUTCHR    NOTE: OUTPUTTING BS PRINTS A
               *                         SPACE & MOVES CURSOR BACK 1
E03F 20 D8              BRA    LINE1


E041 FF                 FCB    $FF,$FF,$FF


E044 7E E0 E3  C1       JMP    CONTRL
               *
E047 74 A0 3D  BADDR    LSR    SWIFLG
E04A 3F                 SWI
E04B 1E                 FCB    HEXIN
E04C 29 F6              BVS    C1
E04E 78 A0 3D  L1       ASL    SWIFLG
E051 39                 RTS


               * "E" COMMAND IN DEBUGGER
E052 3F        ERASCN   SWI
E053 17                 FCB    PFFEED
E054 39                 RTS


E055 8D 53     BYTE     BSR    INHEX
E057 48                 ASL A
E058 48                 ASL A
E059 48                 ASL A
E05A 48                 ASL A
E05B 16                 TAB
E05C 8D 4C              BSR    INHEX
E05E 1B                 ABA
E05F 16                 TAB
E060 39                 RTS


               * JUMP TO USER PROM COMMAND
E061 BD E6 DE  USER     JSR    GETNUM
E064 7E EC 00           JMP    $EC00
               *
E6DE           GETNUM   EQU    $E6DE


E067 44        OUTHL    LSR A
E068 44                 LSR A
E069 44                 LSR A
```

```
E06A 44                 LSR A

E06B 84 0F     OUTHR    AND A  #$0F
E06D 8B 30              ADD A  #$30
E06F 81 39              CMP A  #$39
E071 23 02              BLS    OUTCH
E073 8B 07              ADD A  #$7

E075 7E E1 D1  OUTCH    JMP    OUTEEE

E078 7E E1 AC  INCH     JMP    INEEE

E07B 8D F8     PDATA2   BSR    OUTCH
E07D 08                 INX
E07E A6 00     PDATA1   LDA A  X
E080 81 04              CMP A  #4
E082 26 F7              BNE    PDATA2
E084 39                 RTS

               * ROUTINE TO PROCESS PRINTING CHARACTERS
E085 5D        LINE30   TST B
E086 27 91              BEQ    LINE1
E088 3F                 SWI
E089 11                 FCB    OUTCHR     ECHO CHAR ON SCREEN
E08A A7 00              STA A  X          STORE ENTERED BYTE
E08C 08                 INX
E08D 5A                 DEC B
E08E 20 89              BRA    LINE1      GET NEXT KEY

               * PRINT
               * SEND CHAR IN ACCA TO HARD COPY DEVICE
E090 8D 12     PRINTO   BSR    XPRINT     PRINT THE CHAR
E092 81 0D              CMP A  #CR        TEST FOR CR
E094 26 0D              BNE    PRINT9     IF NOT CR, EXIT
E096 F6 A0 37           LDA B  NULCNT     IF CR, FOLLOW WITH NULLS
E099 27 08              BEQ    PRINT9
E09B 4F                 CLR A
E09C 37        PRINT1   PSH B
E09D 8D 05              BSR    XPRINT
E09F 33                 PUL B
E0A0 5A                 DEC B
E0A1 26 F9              BNE    PRINT1
E0A3 3B        PRINT9   RTI
E0A4 CE 80 00  XPRINT   LDX    #IOBLK
E0A7 7E E1 06           JMP    PDRVRS

E0AA 8D CC     INHEX    BSR    INCH
E0AC 80 30              SUB A  #$30
E0AE 2B 94              BMI    C1
E0B0 81 09              CMP A  #$09
E0B2 2F 0A              BLE    IN1HG
E0B4 81 11              CMP A  #$11
```

```
E0B6 2B 8C                BMI     C1
E0B8 81 16                CMP A   #$16
E0BA 2E 88                BGT     C1
E0BC 80 07                SUB A   #7
E0BE 39          IN1HG    RTS


E0BF A6 00       OUT2H    LDA A   X
E0C1 8D A4       OUT2HA   BSR     OUTHL
E0C3 A6 00                LDA A   X
E0C5 08                   INX
E0C6 20 A3                BRA     OUTHR


E0C8 8D F5       OUT4HS   BSR     OUT2H
E0CA 8D F3       OUT2HS   BSR     OUT2H
E0CC 86 20       OUTS     LDA A   #$20
E0CE 20 A5                BRA     OUTCH


E0D0 74 A0 3D    START    LSR     SWIFLG
E0D3 3F                   SWI
E0D4 0F                   FCB     RESET


E0D5 FF                   FCB     $FF


E0D6 74 A0 3D    SSBENT   LSR     SWIFLG
E0D9 3F                   SWI
E0DA 1F                   FCB     DEBUGR


                 * PBYTE
                 * PRINT THE CONTENTS OF ACCA IN HEXADECIMAL
E0DB 3F          PBYTEO   SWI
E0DC 0D                   FCB     BINHEX     CONVERT BINARY TO HEXADECIMAL
E0DD 3F                   SWI
E0DE 11                   FCB     OUTCHR     PRINT MS DIGIT
E0DF 17                   TBA                MOVE LS DIGIT TO ACCA
E0E0 3F                   SWI
E0E1 11                   FCB     OUTCHR     PRINT LS DIGIT
E0E2 3B                   RTI


E0E3 20 F1       CONTRL   BRA     SSBENT


                 * HEXIN
                 * GET A HEXADECIMAL NUMBER FROM THE KEYBOARD
E0E5 3F          HEXINO   SWI
E0E6 15                   FCB     PSPACE
E0E7 74 A0 3A             LSR     UPCASE     DISABLE LOWER CASE
E0EA 3F                   SWI
E0EB 20                   FCB     LINEIN     INPUT A CHARACTER STRING
E0EC 78 A0 3A             ASL     UPCASE     RE-ENABLE LOWER CASE
E0EF 81 1B                CMP A   #ESC       CHECK FOR ESCAPE
E0F1 26 05                BNE     HEXIN1
E0F3 32                   PUL A              ACCA <-- (CC)
```

```
E0F4 8A 02                 ORA A  #%00000010 SET V-BIT TO INDICATE ESC
E0F6 36                    PSH A             (CC) <-- ACCA
E0F7 3B                    RTI
E0F8 FE A0 1F   HEXIN1     LDX    LINBUF     LOAD BUFFER POINTER
E0FB 3F                    SWI
E0FC 0C                    FCB    HEXBIN     CONVERT STRING TO BINARY
E0FD 3F                    SWI
E0FE 03                    FCB    TBAX       MOVE TO IX ALSO
E0FF 3F                    SWI
E100 0B                    FCB    STAALL     STORE IN STACKED REGISTERS
E101 32                    PUL A
E102 84 FD                 AND A  #%11111101 CLEAR V-BIT TO INDICATE NO ESC
E104 36                    PSH A
E105 3B                    RTI


E106 7D A0 3E   PDRVRS     TST    PRFLAG
E109 26 11                 BNE    SPRINT
E10B A7 12                 STA A  18,X
E10D C6 36                 LDA B  #$36
E10F E7 13                 STA B  19,X
E111 C6 3E                 LDA B  #$3E
E113 E7 13                 STA B  19,X
E115 6D 13      PPR1       TST    19,X
E117 2A FC                 BPL    PPR1
E119 A6 12                 LDA A  18,X
E11B 39                    RTS


E11C E6 0C      SPRINT     LDA B  12,X       GET PRINTER ACIA STATUS
E11E C5 02                 BIT B  #$02       TEST FOR TX EMPTY
E120 27 FA                 BEQ    SPRINT
E122 A7 0D                 STA A  13,X       WHEN TX IS EMPTY STORE CHAR
E124 39                    RTS


                * INECHO
                * ROUTINE FOR KEYBOARD INPUT WITH ECHO
E125 3F         INECHO     SWI
E126 10                    FCB    INCHAR
E127 20 16                 BRA    INCHAX     GO ECHO & STORE CHAR, & EXIT


                * INCHAR
                * KEYBOARD INPUT ROUTINE
E129 7F A0 1C   INCHA0     CLR    ESCSTR
E12C 86 0C                 LDA A  #CBON
E12E 3F                    SWI
E12F 29                    FCB    ESCSND
E130 3F         INCHA1     SWI
E131 26                    FCB    INKEY
E132 27 FC                 BEQ    INCHA1
E134 36                    PSH A
E135 86 0B                 LDA A  #CBOFF
E137 3F                    SWI
E138 29                    FCB    ESCSND
E139 32                    PUL A
E13A 7D A0 38   INCHA3     TST    DUPLEX     TEST FOR FULL DUPLEX/HALF DUPLEX
E13D 27 39                 BEQ    INCHA4     IF FULL DUPLEX, SKIP ECHO
```

```
E13F 3F          INCHAX   SWI
E140 11                   FCB     OUTCHR      IF HALF DUPLEX, ECHO ENTERED CHAR
E141 20 35                BRA     INCHA4      GO STORE DATA IN STACK & EXIT


                 * ESCSND
                 * SEND ESCAPE TO VIDEO DRIVER FOLLOWED BY CODE IN ACCB
E143 16          ESCSNO   TAB
E144 86 1B                LDA A   #ESC
E146 3F                   SWI
E147 11                   FCB     OUTCHR
E148 17                   TBA
E149 3F                   SWI
E14A 11                   FCB     OUTCHR
E14B 3B                   RTI


                 * INKEY
                 * POLL KEYBOARD PIA ONCE FOR CHARACTER
E14C B6 80 11    INKEY0   LDA A   KBDCTL      TEST STATUS FOR DATA READY
E14F 2A 13                BPL     INKEY2      IF B7=0 THEN NO DATA
E151 B6 80 10             LDA A   KBDDAT
E154 7D A0 3A             TST     UPCASE      CHECK FOR FORCE UPPER CASE ON
E157 2B 06                BMI     INKEY1
E159 81 5F                CMP A   #$5F
E15B 23 02                BLS     INKEY1
E15D 80 20                SUB A   #$20        CONVERT LOWER CASE TO UPPER CASE
E15F 30          INKEY1   TSX
E160 A7 02                STA A   2,X         RETURN DATA IN ACCA
E162 20 68                BRA     TBS9        GO SET Z=0 AND EXIT
E164 32          INKEY2   PUL A
E165 8A 04                ORA A   #$04        IF NO DATA, RETURN WITH Z=1
E167 36                   PSH A
E168 3B                   RTI


                 * CASSETTE INPUT ROUTINE
E169 F6 80 00    TAPEIO   LDA B   CASCTL      GET ACIA STATUS
E16C 54                   LSR B               TEST FOR RX FULL
E16D 24 FA                BCC     TAPEIO
E16F B6 80 01             LDA A   CASDAT      GET DATA FROM ACIA
E172 84 7F                AND A   #$7F        CLEAR 8TH BIT
E174 81 7F                CMP A   #$7F        TEST FOR RUBOUT
E176 27 F1                BEQ     TAPEIO      IF RUBOUT, WAIT FOR NEXT CHAR
E178 30          INCHA4   TSX
E179 A7 02                STA A   2,X         RETURN DATA BYTE IN ACCA
E17B 3B                   RTI


                 * INCAPS
                 * INPUT A CHARACTER FORCING UPPER CASE ASCII
E17C 74 A0 3A    INCAPO   LSR     UPCASE      DISABLE LOWER CASE
E17F 3F                   SWI
E180 10                   FCB     INCHAR      INPUT A CHARACTER
E181 78 A0 3A             ASL     UPCASE      RE-ENABLE LOWER CASE
E184 20 F2                BRA     INCHA4


                 * TAPOUT
```

```
                    * CASSETTE OUTPUT ROUTINE
 E186 F6 80 00      TAPOUO   LDA B  CASCTL     GET ACIA STATUS
 E189 C5 02                  BIT B  #2         TEST FOR TX EMPTY
 E18B 27 F9                  BEQ    TAPOUO
 E18D B7 80 01               STA A  CASDAT
 E190 3B                     RTI


 E1AC                        ORG    $E1AC


 E1AC 74 A0 3D      INEEE    LSR    SWIFLG
 E1AF 3F                     SWI
 E1B0 10                     FCB    INCHAR
 E1B1 78 A0 3D      IN1      ASL    SWIFLG
 E1B4 39                     RTS


                    * TBSRCH
                    * SEARCH TABLE FOR ROUTINE INDICATED BY ACCA
 E1B5 E6 00         TBSRCO   LDA B  X
 E1B7 27 13                  BEQ    TBS9
 E1B9 11                     CBA
 E1BA 27 05                  BEQ    TBS2
 E1BC 08                     INX
 E1BD 08                     INX
 E1BE 08                     INX
 E1BF 20 F4                  BRA    TBSRCO
 E1C1 E6 01         TBS2     LDA B  1,X
 E1C3 A6 02                  LDA A  2,X
 E1C5 30                     TSX
 E1C6 E7 03                  STA B  3,X
 E1C8 A7 04                  STA A  4,X
 E1CA 20 98                  BRA    INKEY2
 E1CC 32            TBS9     PUL A
 E1CD 84 FB                  AND A  #%11111011
 E1CF 36                     PSH A
 E1D0 3B                     RTI


 E1D1 74 A0 3D      OUTEEE   LSR    SWIFLG
 E1D4 3F                     SWI
 E1D5 11                     FCB    OUTCHR
 E1D6 20 D9                  BRA    IN1


                             END

NO ERROR(S) DETECTED
```

SYMBOL TABLE:

ADDAX	0004	ADDBAX	0005	BADDR	E047	BEGIN1	A002	BEGIN2	A014
BELL	0007	BINHEX	000D	BLOCK	00FF	BREAKO	0018	BREAK1	0019
BREAK2	001A	BREAK3	001B	BRKTBL	A024	BS	0008	BYTE	E055
BYTECT	A00B	C1	E044	CASCTL	8000	CASDAT	8001	CBOFF	000B
CBON	000C	CHRSLT	A01B	CKSM	A00A	CONTRL	E0E3	CR	000D
CURSOR	A018	DEBUGR	001F	DUPLEX	A038	DURATN	A03F	END1	A004
END2	A016	ENDLIN	A01D	ENDSCN	F780	EOT	0004	ERASCN	E052

ESC	001B	ESCSNO	E143	ESCSND	0029	ESCSTR	A01C	ESCVEC	A03B
FF	000C	GETNUM	E6DE	HEIGHT	0018	HEXBIN	000C	HEXIN	001E
HEXINO	E0E5	HEXIN1	E0F8	IN1	E1B1	IN1HG	E0BE	INCAPO	E17C
INCAPS	002A	INCH	E078	INCHA0	E129	INCHA1	E130	INCHA3	E13A
INCHA4	E178	INCHAR	0010	INCHAX	E13F	INECHO	E125	INECHO	0025
INEEE	E1AC	INHEX	E0AA	INKEY	0026	INKEY0	E14C	INKEY1	E15F
INKEY2	E164	IO	E000	IOBLK	8000	IRQVEC	A000	KBDCTL	8011
KBDDAT	8010	L1	E04E	LF	000A	LINBUF	A01F	LINE	0050
LINE1	E019	LINE15	E029	LINE20	E032	LINE30	E085	LINEIO	E014
LINEIN	0020	LOAD	E00A	LOWLFT	F730	MBTEMP	A00E	MCONT	A011
MEMPTR	A021	MKREAD	0022	MKWRIT	0021	MMODE	A023	MODE	F901
MOVER	000E	MULTAB	0008	NMIVEC	A006	NULCNT	A037	OUT2H	E0BF
OUT2HA	E0C1	OUT2HS	E0CA	OUT4HS	E0C8	OUTCH	E075	OUTCHR	0011
OUTEEE	E1D1	OUTHL	E067	OUTHR	E06B	OUTPTR	A036	OUTS	E0CC
PBYTE	001D	PBYTEO	E0DB	PCLOCN	A048	PCR	0023	PCRLF	0016
PDATA1	E07E	PDATA2	E07B	PDRVRS	E106	PERIOD	A041	PFFEED	0017
PLF	0024	POWDWN	E005	PPCTL	8013	PPDATA	8012	PPR1	E115
PREGS	001C	PRFLAG	A03E	PRINT	0027	PRINTO	E090	PRINT1	E09C
PRINT9	E0A3	PSHALL	0009	PSPACE	0015	PSTRNG	0014	PULALL	000A
RESET	000F	SCREEN	F000	SCROLL	F901	SCRPAD	A000	SLASH	002F
SP	A008	SPACE	0020	SPCTL	800C	SPDATA	800D	SPRINT	E11C
SSBENT	E0D6	STAALL	000B	STAR	002A	START	E0D0	SUBAX	0006
SUBBAX	0007	SUBTOT	A00A	SWIFLG	A03D	TAPEIO	E169	TAPEIN	0012
TAPOUO	E186	TAPOUT	0013	TBAX	0003	TBS2	E1C1	TBS9	E1CC
TBSRCO	E1B5	TBSRCH	0028	TEMP	A00F	TW	A00F	TXBA	0002
UPCASE	A03A	USER	E061	USRVEC	A034	VCNTRL	F900	WAITON	A039
WRAP	A01A	XBA	0000	XBAX	0001	XCRSR	F902	XHI	A00C
XLO	A00D	XPRINT	E0A4	XTEMP	A012	YCRSR	F903		